3GPP (S1.13) V1.0.11999-3

3GPP FDD, spreading and modulation specification

Reference
xxxx

Keywords
Digital cellular telecommunications system,
Universal Mobile Telecommunication System
(UMTS), UTRAN, 3GPP

ETSI Secretariat

Postal address
F-06921 Sophia Antipolis Cedex - FRANCE

Office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16
Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

X.400
c= fr; a=atlas; p=etsi; s=secretariat

Internet
secretariat@etsi.fr
http://www.etsi.fr

Copyright Notification

Reproduction is only permitted for the purpose of standardization work undertaken within ETSI.
The copyright and the foregoing restrictions extend to reproduction in all media.

© European Telecommunications Standards Institute 1998.
All rights reserved.

Contents

1 Intellectual Property Rights 4
2 Foreword 4
3 Scope 4
3.1 Status 4
4 References 4
5 Definitions and abbreviations 4
6 Uplink spreading and modulation 6
6.1 Overview 6
6.2 Spreading 6
6.2.1 Uplink Dedicated Physical Channels (uplink DPDCH/DPCCH) 6
6.2.2 PRACH 8
6.3 Code generation and allocation 8
6.3.1 Channelization codes 8
6.3.2 Scrambling codes 10
6.3.2.1 Long scrambling code 11
6.3.2.2 Short scrambling code 12
6.3.3 Random access codes 14
6.3.3.1 Preamble spreading code 14
6.3.3.2 Preamble signature 16
6.3.3.3 Channelization codes for the message part 16
6.3.3.4 Scrambling code for the message part 17
6.4 Modulation 17
6.4.1 Modulating chip rate 17
6.4.2 Pulse shaping 17
6.4.3 Modulation 17
7 Downlink spreading and modulation 18
7.1 Spreading 18
7.2 Code generation and allocation 19
7.2.1 Channelization codes 19
7.2.2 Scrambling code 19
7.2.3 Synchronisation codes 21
7.2.3.1 Code Generation 21
7.2.3.2 Code Allocation 21
7.3 Modulation 23
7.3.1 Modulating chip rate 23
7.3.2 Pulse shaping 23
7.3.3 Modulation 23
8 History 24

1 Intellectual Property Rights

IPRs essential or potentially essential to the present deliverable may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members, free of charge**. This can be found in the latest version of the ETSI Technical Report: ETR 314: "Intellectual Property Rights (IPRs); Essential or potentially Essential, IPRs notified to ETSI in respect of ETSI standards". The most recent update of ETR 314, is available on the ETSI web server or on request from the Secretariat.
Pursuant to the ETSI Interim IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in the ETR 314, which are, or may be, or may become, essential to the present document.

2 Foreword

This Technical Report (TR) has been produced by the Special Mobile Group (SMG) of the European Telecommunications Standards Institute (ETSI).
This TR describes spreading and modulation for UTRA Physical Layer FDD mode.
The contents of this TR are subject to continuing work within SMG2 and SMG2 UMTS layer 1 expert group and may change following approval by either of these two groups.
<Editor's note: this boilerplate text should be replaced by 3GPP equivalent.>

3 Scope

This European Telecommunication Report (ETR) describes spreading and modulation for UTRA Physical Layer FDD mode.
Text without revision marks has been approved in the previous SMG2 Layer 1 expert group meetings, while text with revision marks is subject to approval.

3.1 Status

The status of all the sections in this specification is working assumption.

4 References

References may be made to:

a) specific versions of publications (identified by date of publication, edition number, version number, etc.), in which case, subsequent revisions to the referenced document do not apply;
b) publications without mention of a specific version, in which case the latest version applies.

A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

5 Definitions and abbreviations

BCH	Broadcast Control Channel
BER	Bit Error Rate
BS	Base Station
CCPCH	Common Control Physical Channel
DCH	Dedicated Channel
DL	Downlink (Forward link)
DPCH	Dedicated Physical Channel
DPCCH	Dedicated Physical Control Channel
DPDCH	Dedicated Physical Data Channel
DS-CDMA	Direct-Sequence Code Division Multiple Access
FACH	Forward Access Channel
FDD	Frequency Division Duplex
Mcps	Mega Chip Per Second
MS	Mobile Station
OVSF	Orthogonal Variable Spreading Factor (codes)
PCH	Paging Channel
PG	Processing Gain
PRACH	Physical Random Access Channel

RACH	Random Access Channel
RX	Receive
SCH	Synchronisation Channel
SF	Spreading Factor
SIR	Signal-to-Interference Ratio
TDD	Time Division Duplex
TFCI	Transport-Format Combination Indicator
TPC	Transmit Power Control
TX	Transmit
UE	User Equipment
UL	Uplink (Reverse link)

6

7 Uplink spreading and modulation

7.1 Overview

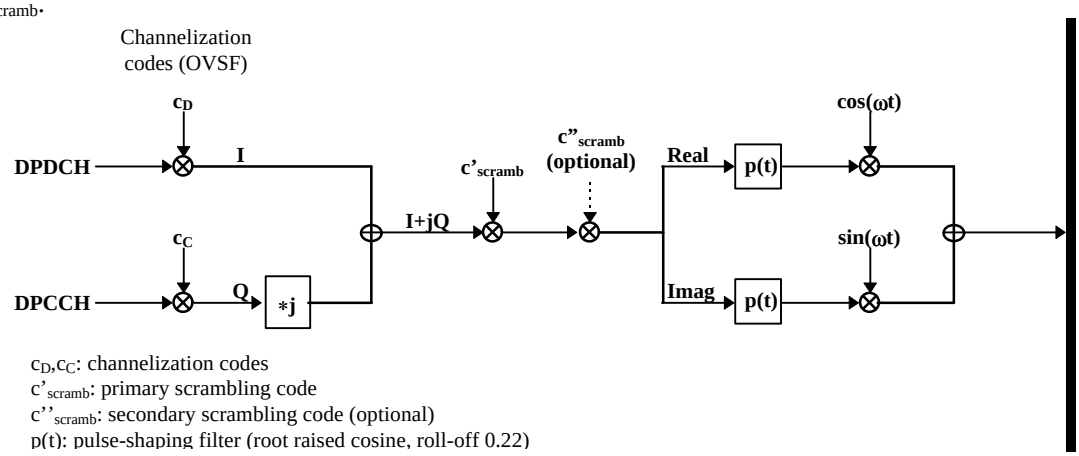
Spreading Modulation is applied after data modulation and before pulse shaping. It consists of two operations. The first is the spreading operation, which transforms every data symbol into a number of chips, thus increasing the bandwidth of the signal. The number of chips per data symbol is called the Spreading Factor (SF). The second operation is the scrambling operation, where a scrambling code is applied to the spread signal.

With the spreading modulation, data symbol on I- and Q-branch are independently multiplied with spreading code. With the scrambling operation, the resultant signals on the I- and Q-branch are further multiplied by complex-valued scrambling code, where I and Q denote real and imaginary parts, respectively. Note that before complex multiplication binary values 0 and 1 are mapped to +1 and -1, respectively.
<Editor's note: the distinction between data modulation and spreading modulation has been included to allow the more comprehensive ARIB text to be included.>

7.2 Spreading

7.2.1 Uplink Dedicated Physical Channels (uplink DPDCH/DPCCH)

Figure 1 illustrates the spreading and modulation for the case of multiple uplink DPDCHs when user services are less than or equal to 1024kbps in the 5MHz band. Note that this figure only shows the principle, and do not necessarily need as they are in an actual implementation. Figure 2 illustrates the case for user services at 2048kbps in the 5 MHz band. Data modulation is dual-channel QPSK (i.e.; separate BPSK on I- and Q-channel), where the uplink DPDCH and DPCCH are mapped to the I and Q branch respectively. The I and Q branch are then spread to the chip rate with two different channelization codes and subsequently complex scrambled by a mobile-station specific complex scrambling code c_{scramb}.



c_D,c_C: channelization codes
c'_{scramb}: primary scrambling code
c''_{scramb}: secondary scrambling code (optional)
p(t): pulse-shaping filter (root raised cosine, roll-off 0.22)

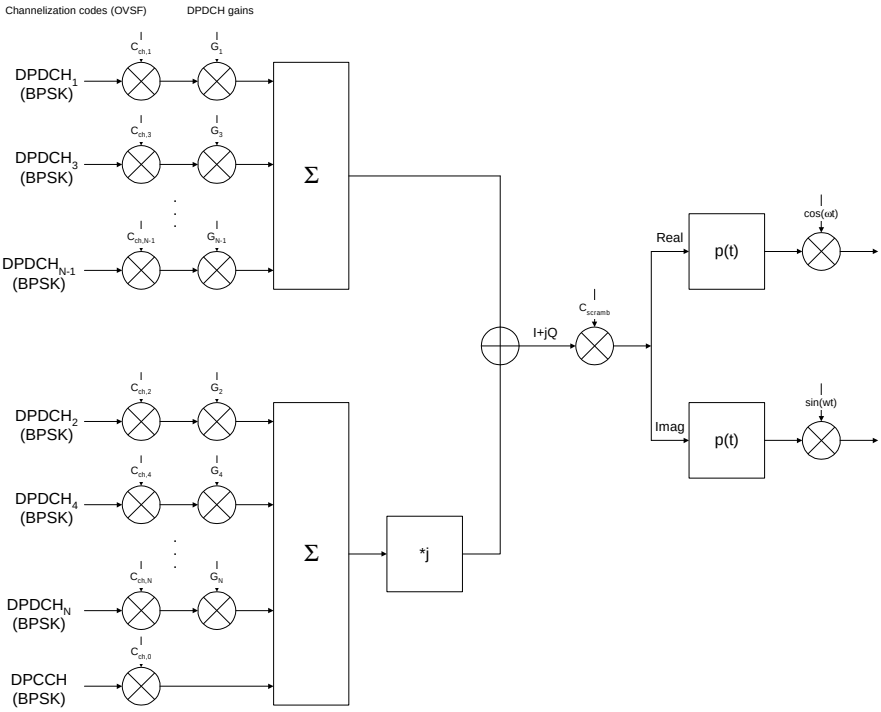
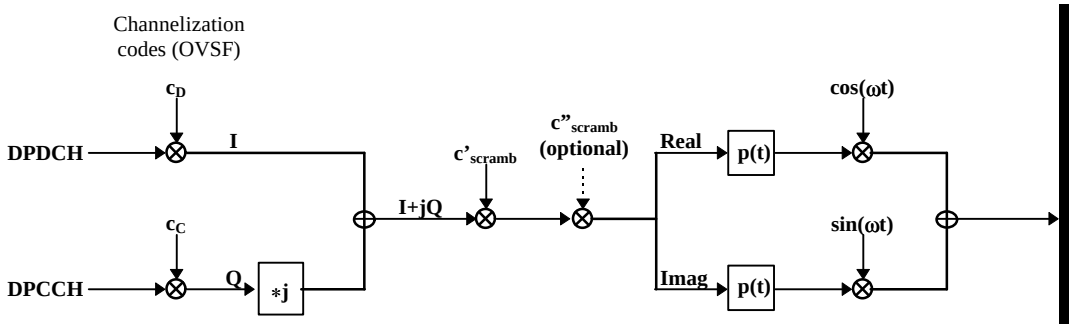


Figure 1 Spreading/modulation for uplink DPDCH/DPCCH for user services less than or equal to 1024kbps in the 5MHz band



c_D,c_C: channelization codes
c'_{scramb}: primary scrambling code
c''_{scramb}: secondary scrambling code (optional)
p(t): pulse-shaping filter (root raised cosine, roll-off 0.22)

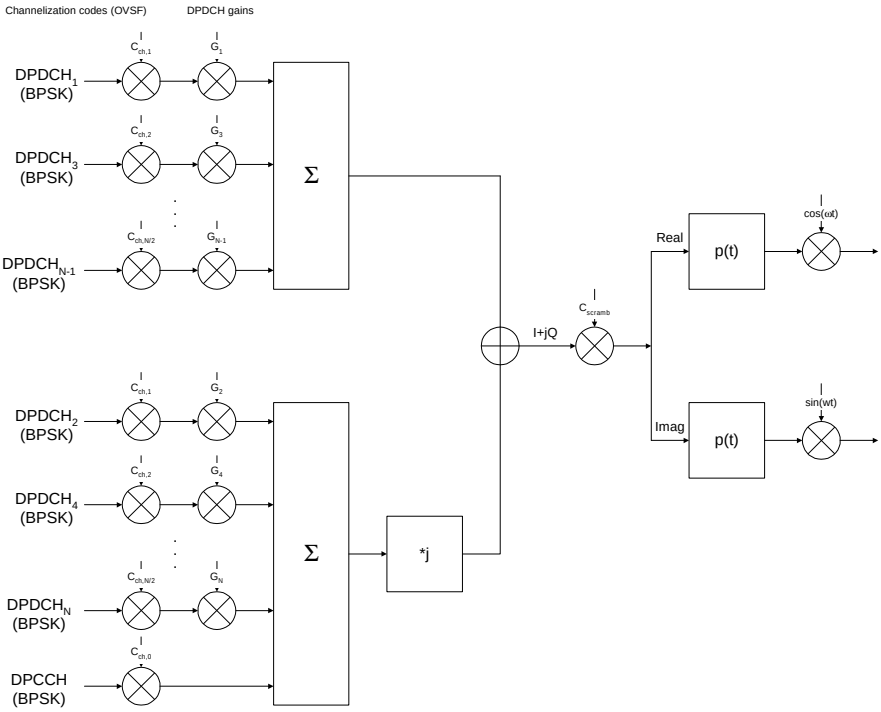


Figure 2. Spreading/modulation for uplink DPDCH/DPCCH for user services at 2048kbps in the 5MHz band

~~<Editor's note: The DPCCH gain 'G' and the order of DPDCH allocations from ARIB are not working assumptions in ETSI. These will be subject to discussion in harmonisation.>~~

<Editor's note: the DPDCH gains were discussed in WG1#2.>

For a single uplink DPDCH transmission, only $DPDCH_1$ and DPCCH are transmitted.]

For services less than or equal to 1024kbps in the 5MHz band, the DPCCH is spread by the channelization code $C_{ch,0}$ and each $DPDCH_i$ is spread by a predefined individual channelization codes, $C_{ch,i}$ (i=1,2,…). For 2048kbps service in the 5MHz band, the DPCCH is spread by the channelization code $C_{ch,0}$ and each pair of $DPDCH_{2i-1}$ and $DPDCH_{2i}$ is spread by a predefined individual channelization codes, $C_{ch,i}$. The data symbols of both the DPDCHs and the DPCCH are BPSK-modulated and the channelization codes are real-valued. The real-valued signals of the I- and Q-branches are then summed and treated as a complex signal. This complex signal is then scrambled by the complex-valued scrambling code, C_{ul_scr}. ~~The power of the DPCCH may be adjusted by a gain factor, G~~. The powers of the DPDCHs may be adjusted by gain factors, G_i.

7.2.2 PRACH

The spreading and modulation of the message part of the Random-Access burst is basically the same as for the uplink dedicated physical channels, see Figure 1, where the uplink DPDCH and uplink DPCCH are replaced by the data part and the control part respectively. The scrambling code for the message part is chosen based on the base-station-specific preamble code.

7.3 Code generation and allocation

7.3.1 Channelization codes

The channelization codes of Figure 1 are Orthogonal Variable Spreading Factor (OVSF) codes that preserve the orthogonality between a user's different physical channels. The OVSF codes can be defined using the code tree of Figure 3.

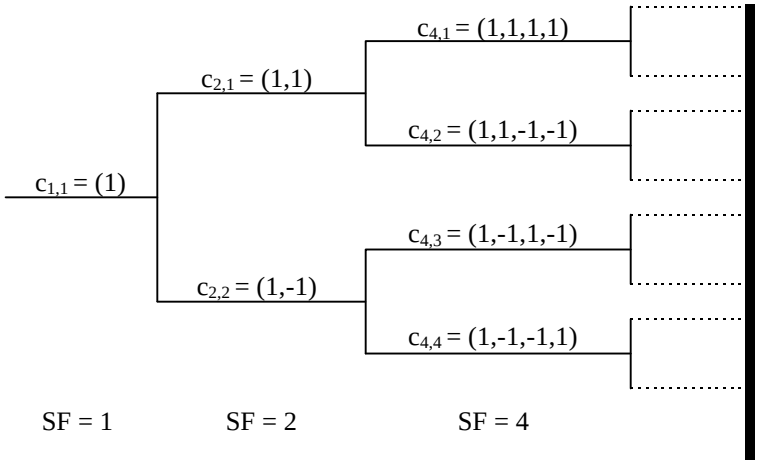


Figure 3. Code-tree for generation of Orthogonal Variable Spreading Factor (OVSF) codes.

In Figure 3, the OVSF code is described as $C_{SF,\text{code number}}$, where $SF_{d,n}$ represents the spreading factor of n^{th} DPDCH. Then the DPCCH is spread by code number 1 with a spreading factor of SF_c.
Each level in the code tree defines channelization codes of length SF, corresponding to a spreading factor of SF in Figure 3. All codes within the code tree cannot be used simultaneously by one mobile station. A code can be used by a mobile station if and only if no other code on the path from the specific code to the root of the tree or in the sub-tree below the specific code is used by the same mobile station. This means that the number of available channelization codes is not fixed but depends on the rate and spreading factor of each physical channel.
The generation method for the channelization code can also be explained in Figure 4.

$$C_0(0) = 1$$

$$\begin{bmatrix} C_1(0) \\ C_1(1) \end{bmatrix} = \begin{bmatrix} C_0(0) & C_0(0) \\ C_0(0) & \overline{C_0(0)} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} C_2(0) \\ C_2(1) \\ C_2(2) \\ C_2(3) \end{bmatrix} = \begin{bmatrix} C_1(0) & C_1(0) \\ C_1(0) & C_1(0) \\ C_1(1) & C_1(1) \\ C_1(1) & C_1(1) \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

$$\vdots$$

$$\begin{bmatrix} C_{n+1}(0) \\ C_{n+1}(1) \\ C_{n+1}(2) \\ C_{n+1}(3) \\ \vdots \\ C_{n+1}(2^{n+1}-2) \\ C_{n+1}(2^{n+1}-1) \end{bmatrix} = \begin{bmatrix} C_n(0) & C_n(0) \\ C_n(0) & C_n(0) \\ C_n(1) & C_n(1) \\ C_n(1) & C_n(1) \\ \vdots & \vdots \\ C_n(2^n-1) & C_n(2^n-1) \\ C_n(2^n-1) & C_n(2^n-1) \end{bmatrix}$$

Figure 4. Spreading Code Generation Method

Binary code words are equivalent to the real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.
The spreading code cycle is the symbol cycle. Thus, for a given chip rate, the spreading code cycle depends on the symbol rate. Furthermore, the number of codes that can be used also differs according to the symbol rate. The relations between symbol rate, spreading code types, spreading code cycle and number of spreading codes is listed in .

The spreading code phase synchronises with the modulation/demodulation symbols. In other words, the head chip of the symbol is spreading code phase=0.

Symbol rate (ksps)				spreading code cycle(chip) SF	No. of Spreading codes
Chip rate= [1.024 Mcps]	4.096 Mcps	[8.192 Mcps]	[16.384 Mcps]		
[1024]	[4096]			1	1
[512]	[2048]	[4096]		2	2
[256]	1024	[2048]	4096]	4	4
[128]	512	[1024]	2048]	8	8
[64]	256	[512]	[1024]	16	16
[32]	128	[256]	[512]	32	32
[16]	64	[128]	[256]	64	64
[8]	32	[64]	[128]	128	128
-	16	[32]	[64]	256	256
-	[8]	[16]	[32]	512	512
-	-	[8]	[16]	1024	1024
			[8]	2048	2048

Table 1. Correspondence between Symbol Rate and Spreading Code Types

<Editor's note: SF=1,2,512 is not adopted by ETSI. Also, ETSI does not support the chip rate of 1.024Mcps and that of 8.192 and 16.384Mcps are ffs.>

Each connection is allocated at least one uplink channelization code, to be used for the uplink DPCCH. In most cases, at least one additional uplink channelization code is allocated for a uplink DPDCH. Further uplink channelization codes may be allocated if more than one uplink DPDCH are required. All channelization codes used for the DPDCHs must be orthogonal to the code used for the DPCCH.

As different mobile stations use different uplink scrambling codes, the uplink channelization codes may be allocated with no co-ordination between different connections. The uplink channelization codes are therefore always allocated in a pre-defined order. The mobile-station and network only need to agree on the number and length (spreading factor) of the uplink channelization codes. The exact codes to be used are then implicitly given. However, for an efficient power amplifier implementation in mobile terminals, the uplink channelization code should be chosen in a manner that reduces the peak to average power ratio. In order to obtain the benefits associated with the scrambling code design at the mobile station transmitter (see Section Scrambling codes

Either short or long scrambling codes should be used on the uplink. The short scrambling code is typically used in cells where the base station is equipped with a an advanced receiver, such as a multi-user detector or interference canceller. With the short scrambling code the cross-correlation properties between different physical channels and users does not vary in time in the same way as when a long code is used. This means that the cross-correlation matrices used in the advanced receiver do not have to be updated as often as for the long scrambling codes case, thereby reducing the complexity of the receiver implementation. In cells where there is no gain in implementation complexity using the short scrambling code, the long code is used instead due to its better interference averaging properties. Both short and long scrambling codes are represented with complex-value.

Alternatively, if the system chooses, RSTS for uplink transmission, the scrambling code is the same as the downlink scrambling code described in 7.2.2. In this case, the same scrambling code is allocated to all dedicated physical channels in the cell.

~~<Editor's note: the HPSK description was moved into the long scrambling code description as the ARIB S(2) code was not specified to use HPSK. If it is decided to use HPSK with S(2) codes then the description will be moved to the general scrambling code section.>~~

Both short and long scrambling codes are formed as follows:

$$Cscramb = c_1(w_0 + jc_2'w_1)$$

where w_0 and w_1 are chip rate sequences defined as repetitions of:

$w_0 = \{1 \quad 1\}$
$w_1 = \{1 \quad -1\}$

where W0 and W1 correspond to $C_{2^n,1}$ and $C_{2^n,2^{n-1}+1}$ of OVSF code shown in Figure 3. where n is SF and n>=1.

Also, c_1 is a real chip rate code, and c_2' is a decimated version of the real chip rate code c_2. The preferred decimation factor is 2, however other decimation factors should be possible in future evolutions of 3GPP if proved desirable.

With a decimation factor N=2, c_2' is given as:

$$c_2'(2k) = c_2'(2k+1) = c_2(2k), \quad k=0,1,2\ldots.$$

These scrambling codes are designed such that at N-1 out of N consecutive chip times they produce +/-90° rotations of the IQ multiplexed data and control channels. At the remaining 1 out of N chip times, they produce 0, +/-90 or 180° rotations . This limits the transitions of the complex baseband signal which is inputted to the root raised cosine pulse shaping filter. This in turn reduces the peak to average ratio of the signal at the filter output, allowing a more efficient power amplifier implementation. To guarantee these desirable properties, restrictions on the choice of uplink OVSF codes are also required (see Section 6.3.1).
~~<Editor's note: the reference 4.2.2.1 in XX.05 made no sense, 6.3.1 is assumed.>~~

The constituent codes c_1 and c_2 are formed differently for the short and long scrambling codes as described in Sections 6.3.2.1 and 6.3.2.2.

7.3.2.1 Long scrambling code

The long uplink scrambling code is typically used in cells without multi-user detection in the base station. The mobile station is informed if a long scrambling code should be used in the Access Grant Message following a Random-Access request and in the handover message.
The long scrambling codes are formed as described in section 6.3.2, where c_1 and c_2 are constructed as the position wise modulo 2 sum ~~[~~of 40960 chip segments~~]~~ of two binary *m*-sequences generated by means of two generator polynomials of degree 41. Let *x*, and *y* be the two *m*-sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^3+X^{41}$. The *y* sequence is constructed using the polynomial $1+ X^{20}+X^{41}$. The resulting sequences thus constitute segments of a set of Gold sequences.
~~*<Editor's note: The above text in [ ] above applies to specifications as adopted by ETSI but not by ARIB.>*~~
The code, c_2, used in generating the quadrature component of the complex spreading code is a [250 μs][875 μs] shifted version of the code, c_1, used in generating the in phase component.
<Editor's note: .there is a choice between the ETSI (250 ms) and ARIB (875 ms) inputs.>

The uplink scrambling code word has a period of one radio frame of ~~[~~10 ms~~][36864 radio frames (2^9 superframes)]. Therefore, the output from the reverse link scrambling code generator shall be terminated at the 36864 radio frame, and the pattern from phase 0 up to the phase for 36864 radio frames is repeated.]~~.
<Editor's note: .there is a choice between the ETSI and ARIB inputs. ETSI supports 10 ms period but ARIB supports 2^9 superframes period. Ad Hoc 10 recommends using 10ms.>
Let $n_{40} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i*:th symbol of the sequence x_n and *y*, respectively
The *m*-sequences x_n and *y* are constructed as:
Initial conditions:
$x_n(0)=n_0$, $x_n(1)= n_1$, $\ldots = x_n(39)= n_{39}$, $x_n(40)= n_{40}$
$y(0)=y(1)= \ldots =y(39)= y(40)=1$
Recursive definition of subsequent symbols:
$x_n(i+41) =x_n(i+3) + x_n(i)$ *modulo 2*, $i=0,\ldots, 2^{41}-43$,
$y(i+41) = y(i+20)+y(i)$ *modulo 2*, $i=0,\ldots, 2^{41}-43$.
The definition of the *n*:th scrambling code word for the in phase and quadrature components follows as (the left most index correspond to the chip scrambled first in each radio frame):
$c_{1,long,n} = < x_n(0)+y(0), x_n(1)+y(1), \ldots, x_n([36864x]N-1)+y([36864x]N-1) >$,

$c_{2,long,n} = < x_n(M)+y(M), x_n(M+1)+y(M+1), \ldots, x_n(M+[36864x]N-1) + y(M+[36864x]N-1) >$,
again all sums being modulo 2 additions. (Both N and M are defined in Table 2.)
<Editor's note: Choice has to be made between 10 ms period or 2^9 super frame period to decide inclusion of [ ].>
The antipodal version of $c_{scr1,n}$, the sequence $c'_{scr1,n}$, is defined as, for $i=0,\ldots, 2^{41}-43$, if $c_{scr1,n}(i)=0$ then $c'_{scr1,n}(i)=1$, and if $c_{scr1,n}(i)=1$ then $c'_{scr1,n}(i)=-1$.
~~The code generator must be able to generate the sequence shifted arbitrarily from the initial state.~~
<Editor's note: It has been suggested in Ad Hoc 10 to remove this restriction.>
~~*<Editor's note: it has been noted as desirable to be able to lift this restriction.>*~~

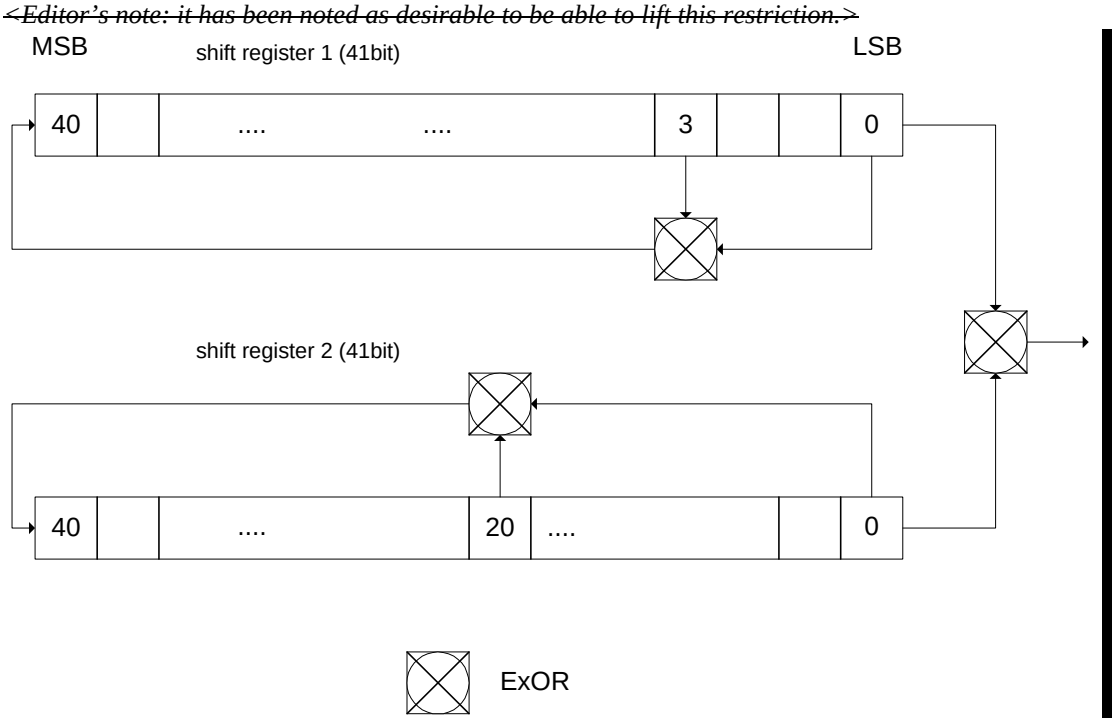


Figure 5. Configuration of uplink scrambling code generator

~~<Editor's note: the above figure is a reconstruction of the ARIB figure due to Word problems.>~~

Chip rate (Mcps)	Period N (chips)	I/Q Offset M (chips)	Range of phase (chip)	
			Primary scrambling code phase (c_1)	Secondary scrambling code phase (c_2')
[1.024	10240	896	0 – [36864x]N-1	M –[36864x]N+(M-1)
4.096	40960	3584		
[8.192	81920	7168		
[16.384	163840	14336		

Table 2. Correspondence between chip rate and uplink scrambling code phase range

<Editor's note: 1.024Mcps is not defined in ETSI document. The chip rate of 8.192and 16.384Mcps are ffs at ETSI.>

7.3.2.2 Short scrambling code

<Editor's note: Ad Hoc 10 has recommended that Short scrambling is not included in Release 99. If this is ratified the short scrambling section will be removed to a suitable place FFS.>
The short scrambling codes are formed as described in Section 6.3.2.1, where, c_1 and c_2 are two different codes from the [extended Very Large Kasami set of length 256] [periodic extended S(2) code family].
<Editor's note:.there is a choice between the ETSI and ARIB inputs. The ETSI scheme has no detailed description for VL Kasami. The following is the ARIB S(2) text.>
[
The reverse link short codes $S_v(n)$, n=0,1,…255, of length 256 chips are obtained by one chip periodic extension of S(2) sequences of length 255. It means that the first chip ($S_v(0)$) and the last chip ($S_v(255)$) of any reverse short scrambling code are the same.
The quaternary S(2) sequence $z_v(n)$, $0 \le v \le 16{,}777{,}216$, of length 255 is obtained by modulo 4 addition of three sequences, a quaternary sequence $a_r(n)$ and two binary sequences $b_s(n)$ and $c_t(n)$, according to the following relation:

$z_v(n) = a_r(n) + 2 \cdot b_s(n) + 2 \cdot c_t(n) \quad (mod\ 4)\ , \quad n = 0, 1, \ldots, 254.$

The user index v determines the indexes r, s, and t of the constituent sequences in the following way:

$v = t \cdot 2^{16} + s \cdot 2^8 + r\ ,$
$r = 0, 1, 2, \ldots, 255,$
$s = 0, 1, 2, \ldots, 255,$
$t = 0, 1, 2, \ldots, 255.$

The quaternary sequence $a_r(n)$ is generated by the recursive generator G_0 defined by the polynomial $g_0(x)= x^8+x^5+3x^3+x^2+2x+1$ as

~~$a_r(n)= a_r(n-3)\ 3\ a_r(n-5)\ a_r(n-6)\ 2\ a_r(n-7)\ a_r(n-8)\ (mod\ 4).$~~
$a_r(n)= 3.a_r(n-3) +1.a_r(n-5) + 3.a_r(n-6) +2.a_r(n-7) + 3.a_r(n-8) \ (\text{mod } 4).$
<Editor's note: this form of the equation has the same meaning as the previous form but is more consistent with the figure below.>

The binary sequence $b_s(n)$ is generated by the recursive generator G_1 defined by the polynomial $g_1(x)= x^8+x^7+x^5+x+1$ as

$b_s(n)= b_s(n-1)+ b_s(n-3)+ b_s(n-7)+ b_s(n-8) \ (mod\ 2).$

The binary sequence $c_t(n)$ is generated by the recursive generator G_2 defined by the polynomial $g_2(x)= x^8+x^7+x^5+x^4+1$ as

$c_t(n)= c_t(n-1)+ c_t(n-3)+ c_t(n-4)+ c_t(n-8) \ (mod\ 2).$

An implementation of the short scrambling code generator is shown in Figure 6. The initial states for the binary generators G_1 and G_2 are the two 8-bit words representing the indexes s and t in the 24-bit binary representation of the user index v, as it is shown in Figure 7.
The initial state for the quaternary generator G_0 is according to Figure 7. obtained after the transformation of 8-bit word representing the index r. This transformation is given by

$a_r(0) = 2v(0)+1 \ (mod\ 4), \quad a_r(n) = 2v(n) \ (mod\ 4), \quad n = 1,\ldots,7.$

The complex quadriphase sequence $S_v(n)$ is obtained from quaternary sequence $z_v(n)$ by the mapping function given in Table 3.
The Re{Sv(n)} and Im{Sv(n)} of the S(2) code are the pair of two binary sequences corresponding to input binary sequences c_1 and c_2 respectively described in 6.3.2.
.

zv(n)	Sv(n)
0	+1 + j1
1	-1 + j1
2	-1 - j1
3	+1 - j1

Table 3. Mapping between $S_v(n)$ and $z_v(n)$

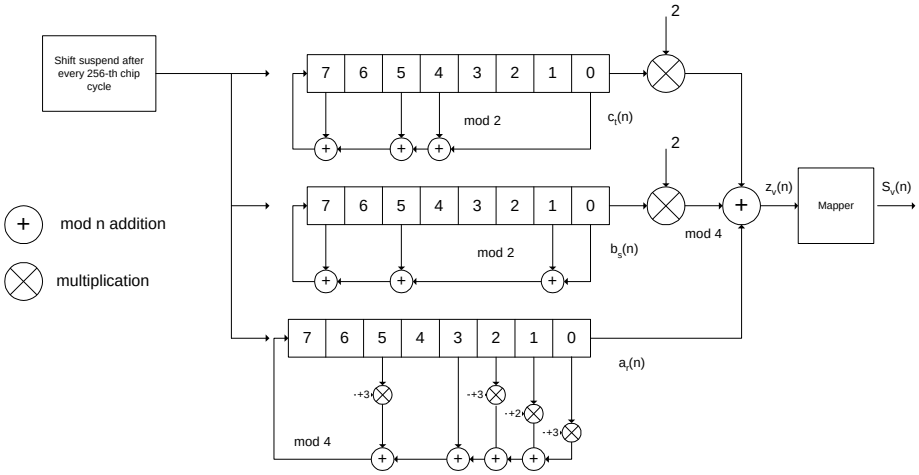


Figure 6. Uplink short scrambling code generator

<editor's note: the above figure has been re-drawn for clarity without changing the meaning.>

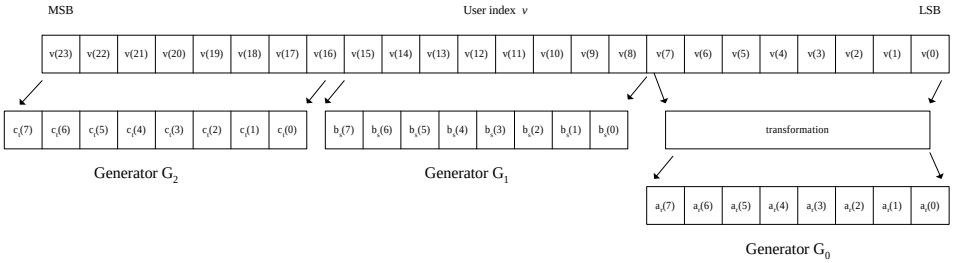


Figure 7. Uplink short scrambling code generator state initialisation

]
The uplink short scrambling code is decided by the network. The mobile station is informed about what short scrambling code to use in the downlink Access Grant message that is the base-station response to an uplink Random Access Request.
The short scrambling code may, in rare cases, be changed during a connection.

7.3.3 Random access codes

<Editor's node: ARIB terminology for PRACH is Reverse Link Common Physical Channel. ARIB has no preamble and the equivalent of the message part has "data" on the I-phase and "signature" on the Q-phase. This is similar to the ETSI message part having data part on I and control part (pilot and rate information) on Q.>

7.3.3.1 Preamble spreading code

The spreading code for the preamble part is cell specific and is broadcast by the base station. More than one preamble code can be used in a base station if the traffic load is high. The preamble codes must be code planned, since two neighbouring cells should not use the same preamble code.

The code used is a real-valued 256 chip Orthogonal Gold code. All 256 codes are used in the system.

The code sequences are constructed with the help of two binary *m*-sequences of length 255, *x,* and *y*, respectively. The *x* sequence is constructed using the polynomial $1+X^2+X^3+X^4+X^8$. The *y* sequence is constructed using the polynomial $1+X^3+X^5+X^6+X^8$.
Let $n_7 \ldots n_0$ be the binary representation of the code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen code number *n* and is denoted x_n in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y,* respectively
The *m*-sequences x_n and *y* are constructed as:
Initial conditions:
$x_n(0)=n_0$, $x_n(1)= n_1$, … $=x_n(6)= n_6$, $x_n(7)= n_7$
$y(0)=y(1)= \ldots =y(6)= y(7)=1$
Recursive definition of subsequent symbols:
$x_n(i+8) =x_n(i+4) + x_n(i+3) + x_n(i+2) + x_n(i)$ *modulo 2, i=0,…,* 246,
$y(i+8) = y(i+6)+ y(i+5)+ y(i+3)+y(i)$ *modulo 2, i=0,…,* 246.

The definition of the *n*:th code word follows (the left most index correspond to the chip transmitted first in each slot):
$C_{RACH,n} = < 0, x_n(0)+y(0), x_n(1)+y(1), \ldots, x_n(254)+y(254) >$,
All sums of symbols are taken modulo 2.
The preamble spreading~~scambling~~ code is described in Figure 8.

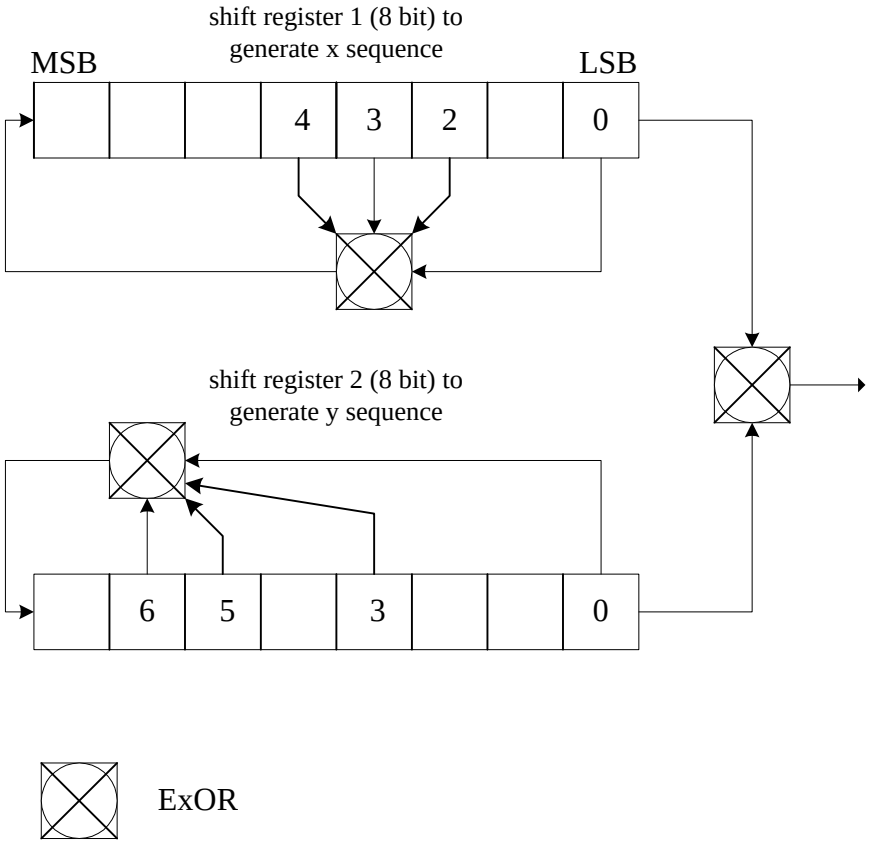


Figure 8. Preamble spreading~~scrambling~~ code generator

Note that the code words always start with a constant '0'. symbol.
Before modulation and transmission these binary code words are converted to real valued sequences by the transformation '0' -> '+1', '1' -> '-1'.

7.3.3.2 Preamble signature

The preamble part carries one of 16 different orthogonal complex signatures of length 16, $<P_0, P_1, ..., P_{15}>$. The signatures are based on a set of Orthogonal Gold codes of length 16 and are specified in Table 4. The base station broadcasts which signatures are allowed to be used in a cell.

	Preamble symbols															
Signature	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
1	A	A	A	-A	-A	-A	A	-A	-A	A	A	-A	A	-A	A	A
2	-A	A	-A	-A	A	A	A	-A	A	A	A	-A	-A	A	-A	A
3	A	-A	A	A	A	-A	A	A	-A	A	A	A	-A	A	-A	A
4	-A	A	-A	A	-A	-A	-A	-A	-A	A	-A	A	-A	A	A	A
5	A	-A	-A	-A	-A	A	A	-A	-A	-A	-A	A	-A	-A	-A	A
6	-A	-A	A	-A	A	-A	A	-A	A	-A	-A	A	A	A	A	A
7	-A	A	A	A	-A	-A	A	A	A	-A	-A	-A	-A	-A	-A	A
8	A	A	-A	-A	-A	-A	-A	A	A	-A	A	A	A	A	-A	A
9	A	-A	A	-A	-A	A	-A	A	A	A	-A	-A	-A	A	A	A
10	-A	A	A	-A	A	A	-A	A	-A	-A	A	A	-A	-A	A	A
11	A	A	A	A	A	A	-A	-A	A	A	-A	A	A	-A	-A	A
12	A	A	-A	A	A	A	A	A	-A	-A	-A	-A	A	A	A	A
13	A	-A	-A	A	A	-A	-A	-A	A	-A	A	-A	-A	-A	A	A
14	-A	-A	-A	A	-A	A	A	A	A	A	A	A	A	-A	A	A
15	-A	-A	-A	-A	A	-A	-A	A	-A	A	-A	-A	A	-A	-A	A
16	-A	-A	A	A	-A	A	-A	-A	-A	-A	A	-A	A	A	-A	A

Table 4. Preamble signatures. A = 1+j.

7.3.3.3 Channelization codes for the message part

The signature in the preamble specifies one of the 16 nodes in the code-tree that corresponds to channelization codes of length 16, as shown in Figure 9. The sub-tree below the specified node is used for spreading of the message part. The control (Q-branch) is spread with the channelization code of spreading factor 256 in the lowest branch of the sub-tree. The data part (I-branch) can use any of the channelization codes from spreading factor 32 to 256 in the upper-most branch of the sub-tree. However, the system may restrict the set of codes (spreading factors) actually allowed in the cell, through the use of a BCH message.

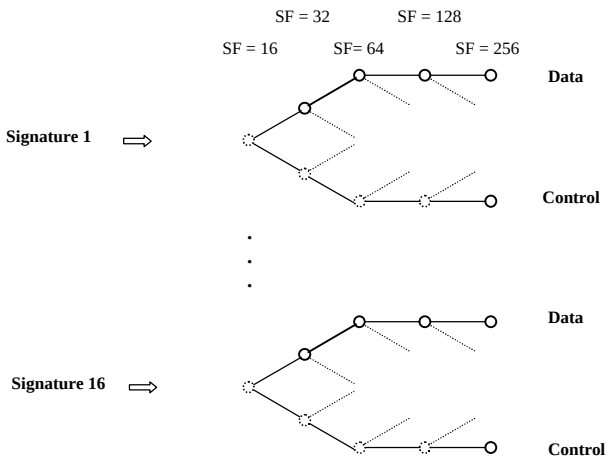


Figure 9. Channelization codes for the random access message part.

Since the control part is always spread with a known channelization code of length 256, it can be detected by the base station. The rate information field of the control part informs the base station about the spreading factor used on the data part. With knowledge of the sub-tree (obtained from the preamble signature) and the spreading factor (obtained from the rate information), the base station knows which channelization code is used for the data part.

This structure allows for simultaneous detection of multiple random access messages arriving in the same access slot, as long as different signatures are used.

7.3.3.4 Scrambling code for the message part

In addition to spreading, the message part is also subject to scrambling with a 10 ms complex code. The scrambling code is cell-specific and has a one-to-one correspondence to the spreading code used for the preamble part. Note that although the scrambling code is the same for every access slot, there is no scrambling-code collision problems between different access slots due to the 1.25 ms time shifts between the access slots.

~~*<Editor's note: the 10ms length for the scrambling code is from ETSI. The ARIB reverse link scrambling scheme has a sequence length of 36,864 frames each of 10 ms. The decision taken for uplink scrambling codes has an effect on the PRACH scrambling codes.>*~~

The scrambling codes used are from the same set of codes as is used for the other dedicated uplink channels when the long scrambling codes are used for these channels. The first 256 of the long scrambling codes are used for the random access channel. The generation of these codes is explained in Section 6.3.2.1. The mapping of these codes to provide a complex scrambling code is also the same as for the other dedicated uplink channels and is described in Section 6.3.2.

7.4 Modulation

7.4.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps.
<Editor's note: 1.024 is in ARIB but not ETSI.>

7.4.2 Pulse shaping

The pulse-shaping filters are root-raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

7.4.3 Modulation

In the uplink, the data modulation of both DPCCH and DPDCH is BPSK. The modulated DPCCH is mapped to the Q-channel, while the first DPDCH is mapped to the I-channel. Subsequently added DPDCHs are mapped alternatively to the I or Q-channel.

8 Downlink spreading and modulation

8.1 Spreading

Figure 10 illustrates the spreading and modulation for the downlink DPCH and CCPCHs. Data modulation is QPSK where each pair of two bits are serial-to-parallel converted and mapped to the I and Q branch respectively. The I and Q branch are then spread to the chip rate with the same channelization code c_{ch} (real spreading) and subsequently scrambled by the same cell specific scrambling code C_{scramb} (complex scrambling).

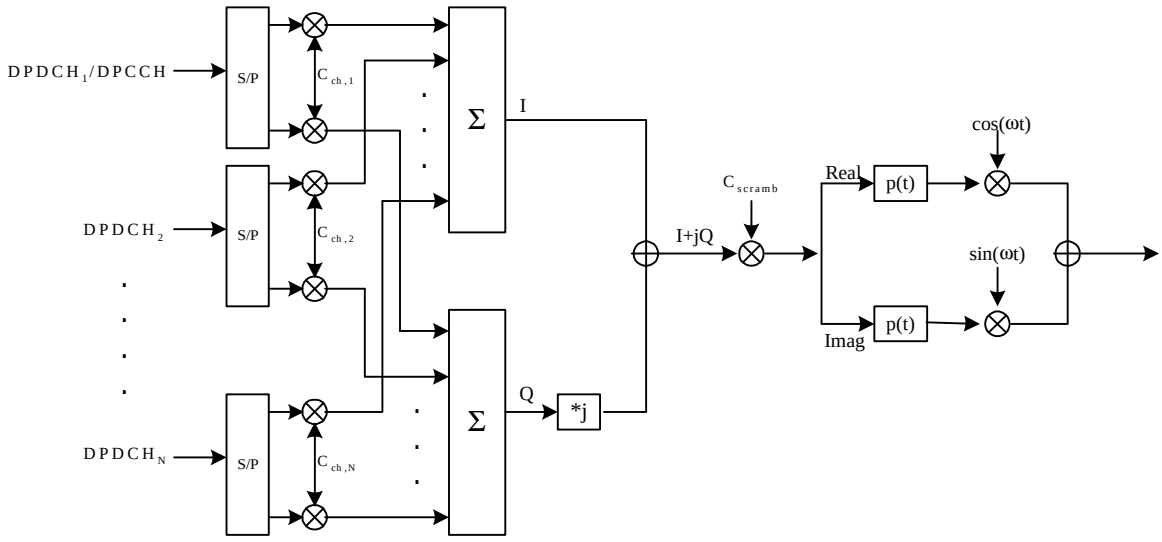


Figure 10. Spreading/modulation for downlink DPCH and CCPCHs.

The different physical channels use different channelization codes, while the scrambling code is the same for all physical channels in one cell.

The multiplexing of the [SCH]{1st and 2nd search codes} with [Primary CCPCH] {Perch channel} is illustrated in Figure 11. The figure illustrates that the [SCH]{1st and 2nd search code} are transmitted alternately. [Primary SCH and Secondary SCH] {The first search code and the second search code} are code multiplexed and transmitted simultaneously during the 1st symbol (256 chips) of each slot. In a frame. The transmission power of [SCH] {search code symbol} can be adjusted by a gain factor $G_{P\text{-}SCH}$ and $G_{S\text{-}SCH}$, respectively, independent of transmission power of P-CCPCH. The [SCH]{search code} is *non-orthogonal* to the other downlink physical channels.

[Editor's note: Take texts in [ ] if ETSI's SCH is chosen. Take texts in { } if ARIB's perch channel scheme is chosen.]

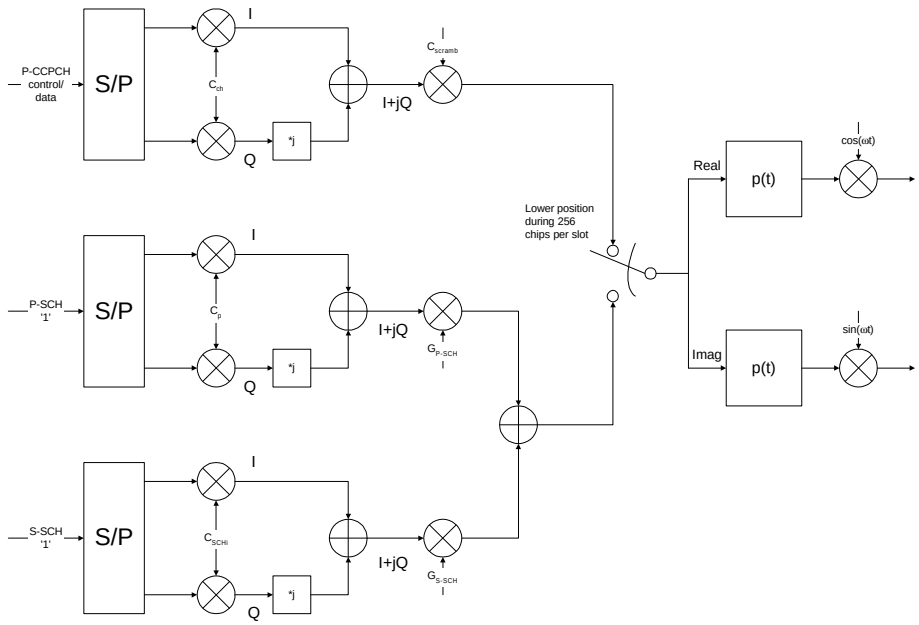


Figure 11. [Multiplexing of SCH]{Transmission of PCCPCH}

<Editor's note: the optional scale parameters reflects the suggestion from the Ad Hoc that CCPCH and SCH channels need not have the same amplitude. If this is not adopted the scale factor will be unity.>~~<Editor's note: The above two figures should be drawn in a more consistent way, maybe as one figure only. >~~

8.2 Code generation and allocation

8.2.1 Channelization codes

The channelization codes of Figure 10 are the same codes used in the uplink, namely Orthogonal Variable Spreading Factor (OVSF) codes that preserve the orthogonality between downlink channels of different rates and spreading factors. The OVSF codes are defined in Figure 3 in Section 6.3.1. The same restriction on code allocation applies as for the uplink, but for a cell and not a mobile station as in the uplink. Hence, in the downlink a code can be used in a cell if and only if no other code on the path from the specific code to the root of the tree or in the sub-tree below the specific code is used in the same cell.

The channelization code for the BCH is a predefined code which is the same for all cells within the system.

The channelization code(s) used for the [Secondary Common Control Physical Channel][Common Physical Channel] is broadcast on the BCH.

< Editor's note: There is a choice in terminology.>

The channelization codes for the downlink dedicated physical channels are decided by the network. The mobile station is informed about what downlink channelization codes to receive in the downlink Access Grant message that is the base-station response to an uplink Random Access request. The set of channelization codes may be changed during the duration of a connection, typically as a result of a change of service or an inter-cell handover. A change of downlink channelization codes is negotiated over a DCH.

8.2.2 Scrambling code

The total number of available scrambling codes is 512, divided into 32 code groups with 16 codes in each group. The grouping of the downlink codes is done in order to facilitate a fast cell search, see document S1.14. The downlink scrambling code is assigned to the cell (sector) at the initial deployment. The mobile station learns about the downlink scrambling code during the cell search process, see document S1.14.

~~*<Editor's note: this dependency on S1.14 should be verified during harmonisation>.*~~

[In order to avoid code limitation in some cases, e.g. when increasing the capacity using adaptive antennas, the possibility to associate several scrambling codes with one cell (BCH area) has been identified as one solution. The exact implementation of such a scheme is still to be determined.]

<Editor's note: Use of multiple downlink scrambling codes to aid adaptive antennas are not specified in ARIB document and treated as ffs.>

The scrambling code sequences are constructed by combining two real sequences into a complex sequence. Each of the two real sequences are constructed as the position wise modulo 2 sum of [40960 chip segments of] two binary *m*-sequences generated by means of two generator polynomials of degree 18. The resulting sequences thus constitute segments of a set of Gold sequences. The scrambling codes are repeated for every 10 ms radio frame. Let *x* and *y* be the

two sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^7+X^{18}$. The y sequence is constructed using the polynomial $1+X^5+X^7+X^{10}+X^{18}$.
*<Editor's note: [ ] is due to the fact that only 4.096Mcps is defined in the ETSI documents. ARIB documents also defines 1.024, 8.196, and 16.384Mcps.>*Let $n_{17} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y*, respectively
The *m*-sequences x_n and *y* are constructed as:
Initial conditions:
$x_n(0)=n_0$, $x_n(1)= n_1$, $\ldots = x_n(16)= n_{16}$, $x_n(17)= n_{17}$
$y(0)=y(1)= \ldots =y(16)= y(17)=1$
Recursive definition of subsequent symbols:
$x_n(i+18) = x_n(i+7) + x_n(i)$ *modulo 2,* $i=0,\ldots,2^{18}-20$,
$y(i+18) = y(i+10)+y(i+7)+y(i+5)+y(i)$ *modulo 2,* $i=0,\ldots, 2^{18}-20$.
The n:th Gold code sequence z_n is then defined as
$z_n(i) = x_n(i) + y(i)$ *modulo 2,* $i=0,\ldots, 2^{18}-2$.
The antipodal version of z_n, the sequence z'_n, is defined as, for $i=0,\ldots, 2^{18}-2$, if $z_n(i)=0$ *then* $z'_n(i)=1$, *and if* $z_n(i)=1$ *then* $z'_n(i)=-1$.
Finally, the n:th complex scrambling code sequence C_{scramb} is defined as (the lowest index corresponding to the chip scrambled first in each radio frame): (see Table 5 for definition of N and M)

$C_{scramb}(i) = z'_n(i) + j\, z'_n(i+M)$, $i=0,1,\ldots,N-1$.
<Editor's note: the values 3584 and 40960 are based on an assumption of a chip rate of 4.096 Mcps.>
Note that the pattern from phase 0 up to the phase of 10 msec is repeated.
The index *n* runs from 0 to 511 giving 512 distinct 40960 chip sequences.
<Editor's note: in ARIB the generator is required to be able to realise a clock arbitrarily shifted from the initial phase. It has been suggested in Ad Hoc 10 to not include this restriction.>

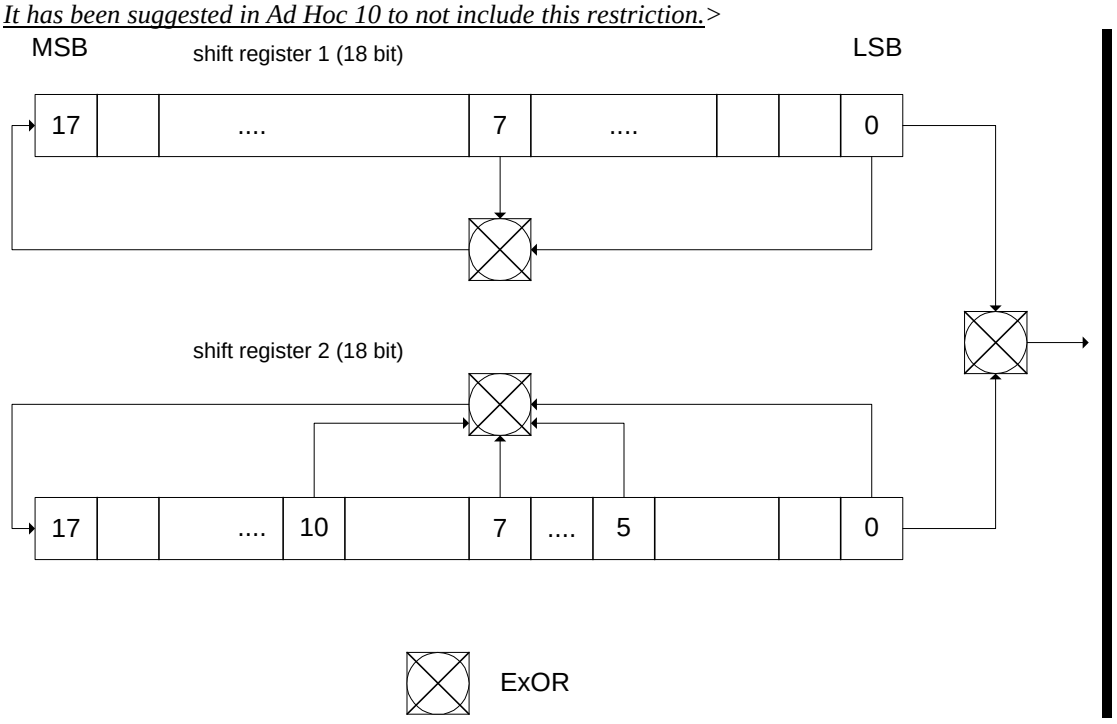


Figure 12. Configuration of downlink scrambling code generator

<Editor's note: the above figure is a reconstruction of the ARIB figure due to Word problems.>

chip rate (Mcps)	Period N	IQ Offset M	Range of phase (chip)	
			for in-phase component	for quadrature component
[1.024	[10240]	[896]	0 – N-1	M – N+M-1
4.096	40960	3584		
[8.192	[81920]	[7168]		
[16.384	[163840]	[14336]		

Table 5. Correspondence between chip rate and downlink scrambling code phase range

8.2.3 Synchronisation codes

8.2.3.1 Code Generation

The Primary and Secondary code words, C_p and $\{C_1,\ldots,C_{17}\}$ are constructed as the position wise addition modulo 2 of a Hadamard sequence and a fixed so called hierarchical sequence. The [Primary SCH] [1st search code] is furthermore chosen to have good aperiodic auto correlation properties.

< Editor's note: There is a choice on the terminology. Also, the text in the 2nd [ ] needs to be verified>

The hierarchical sequence y sequence is constructed from two constituent sequences x_1 and x_2 of length n_1 and n_2 respectively using the following formula:

$y(i) = x_2(i \bmod n_2) + x_1(i \operatorname{div} n_2)$ *modulo 2*, $i = 0 \ldots (n_1 * n_2) - 1$

The constituent sequences x_1 and x_2 are chosen to be identical and to be the following length 16 (i.e. $n_1 = n_2 = 16$) sequence:

$x_1 = x_2 = < 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0 >$

The Hadamard sequences are obtained as the rows in a matrix H_8 constructed recursively by:

$$H_0 = (0)$$

$$H_k = \begin{pmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & \overline{H_{k-1}} \end{pmatrix}, \quad k \geq 1$$

The rows are numbered from the top starting with row *0* (the all zeros sequence).

The Hadamard sequence *h* depends on the chosen code number *n* and is denoted h_n in the sequel.

[This code number is chosen from every 8th low of the matrix H_8. Therefore, there are 32 possible code numbers out of which 17 are used.]

<Editor's note: Only ARIB input specifies this code group out of which 17 codes are chosen but it has to do with fast Hadamard transformation>

Furthermore, let $h_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence h_n and *y*, respectively.

$h(i)$ is identical to $C_{2^8,i'}$ where i' is the bit-reversed number of the 8-bit binary representation of i.

The definition of the *n*:th [SCH][search] code word follows (the left most index correspond to the chip transmitted first in each slot):

$C_{SCH,n} = < h_n(0)+y(0), h_n(1)+y(1), h_n(2)+y(2), \ldots, h_n(255)+y(255) >$,

All sums of symbols are taken modulo 2.

Before modulation and transmission these binary code words are converted to real valued sequences by the transformation '0' -> '+1', '1' -> '-1'.

The [Primary SCH][1st] search] and [Secondary SCH][2nd search] code words are defined in terms of $C_{SCH,n}$ and the definition of C_p and $\{C_1,\ldots,C_{17}\}$ now follows as:

$C_p = C_{SCH,0}$

and

$C_i = C_{SCH,i}$, i=1,…,17

The definitions of C_p and $\{C_1,\ldots,C_{17}\}$ are such that a 32 point fast Hadamard transform can be utilised for detection.

<Editor's note: choice has to be made between for example primary SCH code and 1st search code.>

8.2.3.2 Code Allocation

<Editor's note: Descriptions in ETSI's XX.03 5.3.2.3 and in ARIB's 3.2.4.2.3.1.3.1. can be found here.>

<Editor's note: Following text is copied in large part out of 5.3.2.3 of S1.11 with insertion of text taken out of ARIB document where appropriate as the code allocation of Primary and Secondary SCH channels (or 1st and 2nd search code according to ARIB terminology) is more suitably found in this section. (See editorial comment in 5.3.2.3 of S1.11>

The 32 sequences are constructed such that their cyclic-shifts are unique, i.e., a non-zero cyclic shift less than 16 of any of the 32 sequences is not equivalent to some cyclic shift of any other of the 32 sequences. Also, a non-zero cyclic shift less than 16 of any of the sequences is not equivalent to itself with any other cyclic shift less than 16. This property is

used to uniquely determine both the long code group and the frame timing in the second step of acquisition (see S1.14 Section 4.1). The following sequences are used to encode the 32 different code groups each containing 16 scrambling codes (note that c_i indicates the i'th Secondary Short code [2nd search code] of the 17 codes). Note that a Secondary Short code [2nd search code] can be different from one time slot to another and that the sequence pattern can be different from one cell to another, depending on Scrambling Code Group of Scrambling Code the cell uses~~. [In this table, the last pattern (33rd : Sync BTS) is assigned to the synchronous base station operations, in which only one Scrambling Code is used.]~~

~~*<Editor's note: the 33rd sequence in the table below is used for detecting synchronous CDMA systems such as cdma2000. As such, the 33rd sequence is only found in ARIB document.>*~~

Scrambling Code Groups	Slot Number															
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
Group1	C_1	C_1	C_2	C_{11}	C_6	C_3	C_{15}	C_7	C_8	C_8	C_7	C_{15}	C_3	C_6	C_{11}	C_2
Group2	C_1	C_2	C_9	C_3	C_{10}	C_{11}	C_{13}	C_{13}	C_{11}	C_{10}	C_3	C_9	C_2	C_1	C_{16}	C_{16}
Group 3	C_1	C_3	C_{16}	C_{12}	C_{14}	C_2	C_{11}	C_2	C_{14}	C_{12}	C_{16}	C_3	C_1	C_{13}	C_4	C_{13}
Group 4	C_1	C_4	C_6	C_4	C_1	C_{10}	C_9	C_8	C_{17}	C_{14}	C_{12}	C_{14}	C_{17}	C_8	C_9	C_{10}
Group 5	C_1	C_5	C_{13}	C_{13}	C_5	C_1	C_7	C_{14}	C_3	C_{16}	C_8	C_8	C_{16}	C_3	C_{14}	C_7
Group 6	C_1	C_6	C_3	C_5	C_9	C_9	C_5	C_3	C_6	C_1	C_4	C_2	C_{15}	C_{15}	C_2	C_4
Group 7	C_1	C_7	C_{10}	C_{14}	C_{13}	C_{17}	C_3	C_9	C_9	C_3	C_{17}	C_{13}	C_{14}	C_{10}	C_7	C_1
Group 8	C_1	C_8	C_{17}	C_6	C_{17}	C_8	C_1	C_{15}	C_{12}	C_5	C_1	C_7	C_{13}	C_5	C_{12}	C_{15}
Group 9	C_1	C_9	C_7	C_{15}	C_4	C_{16}	C_{16}	C_4	C_{15}	C_7	C_9	C_1	C_{12}	C_{17}	C_{17}	C_{12}
Group 10	C_1	C_{10}	C_{14}	C_7	C_8	C_7	C_{14}	C_{10}	C_1	C_9	C_5	C_{12}	C_{11}	C_{12}	C_5	C_9
Group 11	C_1	C_{11}	C_4	C_{16}	C_{12}	C_{15}	C_{12}	C_{16}	C_4	C_{11}	C_1	C_6	C_{10}	C_7	C_{10}	C_6
Group 12	C_1	C_{12}	C_{11}	C_8	C_{16}	C_6	C_{10}	C_5	C_7	C_{13}	C_{14}	C_{17}	C_9	C_2	C_{15}	C_3
Group 13	C_1	C_{13}	C_1	C_{17}	C_3	C_{14}	C_8	C_{11}	C_{10}	C_{15}	C_{10}	C_{11}	C_8	C_{14}	C_3	C_{17}
Group 14	C_1	C_{14}	C_8	C_9	C_7	C_5	C_6	C_{17}	C_{13}	C_{17}	C_6	C_5	C_7	C_9	C_8	C_{14}
Group 15	C_1	C_{15}	C_{15}	C_1	C_{11}	C_{13}	C_4	C_6	C_{16}	C_2	C_2	C_{16}	C_6	C_4	C_{13}	C_{11}
Group 16	C_1	C_{16}	C_5	C_{10}	C_{15}	C_4	C_2	C_{12}	C_2	C_4	C_{15}	C_{10}	C_5	C_{16}	C_1	C_8
Group 17	C_1	C_{17}	C_{12}	C_2	C_2	C_{12}	C_{17}	C_1	C_5	C_6	C_{11}	C_4	C_4	C_{11}	C_6	C_5
Group 18	C_2	C_8	C_{11}	C_{15}	C_{14}	C_1	C_4	C_{10}	C_{10}	C_4	C_1	C_{14}	C_{15}	C_{11}	C_8	C_2
Group 19	C_2	C_9	C_1	C_7	C_1	C_9	C_2	C_{16}	C_{13}	C_6	C_{14}	C8	C_{14}	C_6	C_{13}	C_{16}
Group 20	C_2	C_{10}	C_8	C_{16}	C_5	C_{17}	C_{17}	C_5	C_{16}	C_8	C_{10}	C_2	C_{13}	C_1	C_1	C_{13}
Group 21	C_2	C_{11}	C_{15}	C_8	C_9	C_8	C_{15}	C_{11}	C_2	C_{10}	C_6	C_{13}	C_{12}	C_{13}	C_6	C_{10}
Group 22	C_2	C_{12}	C_5	C_{17}	C_{13}	C_{16}	C_{13}	C_{17}	C_5	C_{12}	C_2	C_7	C_{11}	C_8	C_{11}	C_7
Group 23	C_2	C_{13}	C_{12}	C_9	C_{17}	C_7	C_{11}	C_6	C_8	C_{14}	C_{15}	C_1	C_{10}	C_3	C_{16}	C_4
Group 24	C_2	C_{14}	C_2	C_1	C_4	C_{15}	C_9	C_{12}	C_{11}	C_{16}	C_{11}	C_{12}	C_9	C_{15}	C_4	C_1
Group 25	C_2	C_{15}	C_9	C_{10}	C_8	C_6	C_7	C_1	C_{14}	C_1	C_7	C_6	C_8	C_{10}	C_9	C_{15}
Group 26	C_2	C_{16}	C_{16}	C_2	C_{12}	C_{14}	C_5	C_7	C_{17}	C_3	C_3	C_{17}	C_7	C_5	C_{14}	C_{12}
Group 27	C_2	C_{17}	C_6	C_{11}	C_{16}	C_5	C_3	C_{13}	C_3	C_5	C_{16}	C_{11}	C_6	C_{17}	C_2	C_9
Group 28	C_2	C_1	C_{13}	C_3	C_3	C_{13}	C_1	C_2	C_6	C_7	C_{12}	C_5	C_5	C_{12}	C_7	C_6
Group 29	C_2	C_2	C_3	C_{12}	C_7	C_4	C_{16}	C_8	C_9	C_9	C8	C_{16}	C_4	C_7	C_{12}	C_3
Group 30	C_2	C_3	C_{10}	C_4	C_{11}	C_{12}	C_{14}	C_{14}	C_{12}	C_{11}	C_4	C_{10}	C_3	C_2	C_{17}	C_{17}
Group 31	C_2	C_4	C_{17}	C_{13}	C_{15}	C_3	C_{12}	C_3	C_{15}	C_{13}	C_{17}	C_4	C_2	C_{14}	C_5	C_{14}
Group 32	C_2	C_5	C_7	C_5	C_2	C_{11}	C_{10}	C_9	C_1	C_{15}	C_{13}	C_{15}	C_1	C_9	C_{10}	C_{11}
[SyncBTS]	C_2	C_6	C_{14}	C_{14}	C_6	C_2	C_8	C_{15}	C_4	C_{17}	C_9	C_9	C_{17}	C_4	C_{15}	C_8
~~[SyncBTS]~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~

Table 9 Spreading Code allocation for Secondary SCH [Second Search] Code
<Editor's note: The ARIB document assigns the sequence number to each time slot but the ETSI document does not.>

8.3 Modulation

8.3.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps.
<Editor's note: 1.024 is in ARIB but not ETSI.>

8.3.2 Pulse shaping

The pulse-shaping filters are root raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

8.3.3 Modulation

QPSK modulation is used.

9 History

Document history		
draft	1999-02-12	New document merged from ETSI XX.05 and ARIB 3.2.4 sources.
0.0.1	1999-02-12	Corrected typo in table2.
0.0.2	1999-02-16	Added sec. SCH code table, option for HPSK on S(2) codes, scale on SCH.
0.0.3	1999-02-18	Reflected decision made on SCH multiplexing (see document titled 'Report from Ad Hoc #2 SCH multiplexing'.) and additional description on the use of S(2) for uplink short scrambling code.
0.1.0	1999-02-28	Raised to 0.1.0 after TSG RAN WG1#2 meeting (Yokohama).
1.0.0	1999-03-12	Raised to 1.0.0 when presented to TSG RAN.
1.0.1	1999-03-17	Raised to 1.0.1 incorporated Ad Hoc changes and errata from e-mail.

Editor for S1.13, spreading and modulation description, is:

Peter Chambers
Siemens Roke Manor Research
Email: peter.chambers@roke.co.uk

This document is written in Microsoft Word 97.

), the OVSF codes should be selected as follows:
The DPCCH is spread by code number 0 in any code tree as described in Section Channelization codes. The first DPDCH is spread by code number ($SF_{d,1}$ / 4 + 1). Subsequently added DPDCHs for multi-code transmission are spread by codes in ascending order starting from code number 2 excepting the one used for the first DPDCH. However to guarantee the orthogonality between channels, any subtree below the specified node is not used for the channelization code of a DPDCH.
(Note: The case of OVSF code allocation with multiple DPDCHs with different spreading factors is for further study)

7.3.2 Scrambling codes

Either short or long scrambling codes should be used on the uplink. The short scrambling code is typically used in cells where the base station is equipped with a an advanced receiver, such as a multi-user detector or interference canceller. With the short scrambling code the cross-correlation properties between different physical channels and users does not vary in time in the same way as when a long code is used. This means that the cross-correlation matrices used in the advanced receiver do not have to be updated as often as for the long scrambling codes case, thereby reducing the complexity of the receiver implementation. In cells where there is no gain in implementation complexity using the short scrambling code, the long code is used instead due to its better interference averaging properties. Both short and long scrambling codes are represented with complex-value.

Alternatively, if the system chooses, RSTS for uplink transmission, the scrambling code is the same as the downlink scrambling code described in Scrambling code. In this case, the same scrambling code is allocated to all dedicated physical channels in the cell.
~~<Editor's note: the HPSK description was moved into the long scrambling code description as the ARIB S(2) code was not specified to use HPSK. If it is decided to use HPSK with S(2) codes then the description will be moved to the general scrambling code section.>~~

Both short and long scrambling codes are formed as follows:

$$Cscramb = c_1(w_0 + jc_2'w_1)$$

where w_0 and w_1 are chip rate sequences defined as repetitions of:

w_0 = {1 1}
w_1 = {1 -1}

where W0 and W1 correspond to $C_{2^n,1}$ and $C_{2^n,2^{n-1}+1}$ of OVSF code shown in Figure 3. where n is SF and n>=1.

Also, c_1 is a real chip rate code, and c_2' is a decimated version of the real chip rate code c_2. The preferred decimation factor is 2, however other decimation factors should be possible in future evolutions of 3GPP if proved desirable.

With a decimation factor N=2, c_2' is given as:

$$c_2'(2k) = c_2'(2k+1) = c_2(2k), \quad k=0,1,2\ldots.$$

These scrambling codes are designed such that at N-1 out of N consecutive chip times they produce +/-90° rotations of the IQ multiplexed data and control channels. At the remaining 1 out of N chip times, they produce 0, +/-90 or 180° rotations . This limits the transitions of the complex baseband signal which is inputted to the root raised cosine pulse shaping filter. This in turn reduces the peak to average ratio of the signal at the filter output, allowing a more efficient power amplifier implementation. To guarantee these desirable properties, restrictions on the choice of uplink OVSF codes are also required (see Section Channelization codes).
~~<Editor's note: the reference 4.2.2.1 in XX.05 made no sense, 6.3.1 is assumed.>~~

The constituent codes c_1 and c_2 are formed differently for the short and long scrambling codes as described in Sections Long scrambling code and Short scrambling code.

7.3.2.1 Long scrambling code

The long uplink scrambling code is typically used in cells without multi-user detection in the base station. The mobile station is informed if a long scrambling code should be used in the Access Grant Message following a Random-Access request and in the handover message.
The long scrambling codes are formed as described in section 6.3.2, where c_1 and c_2 are constructed as the position wise modulo 2 sum ~~[~~of 40960 chip segments~~]~~ of two binary *m*-sequences generated by means of two generator polynomials of degree 41. Let *x*, and *y* be the two *m*-sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^3+X^{41}$. The *y* sequence is constructed using the polynomial $1+ X^{20}+X^{41}$. The resulting sequences thus constitute segments of a set of Gold sequences.
~~*<Editor's note: The above text in [ ] above applies to specifications as adopted by ETSI but not by ARIB.>*~~
The code, c_2, used in generating the quadrature component of the complex spreading code is a [250 μs][875 μs] shifted version of the code, c_1, used in generating the in phase component.
<Editor's note: .there is a choice between the ETSI (250 ms) and ARIB (875 ms) inputs.>

The uplink scrambling code word has a period of one radio frame of ~~[~~10 ms~~][36864 radio frames (2^9 superframes)]. Therefore, the output from the reverse link scrambling code generator shall be terminated at the 36864 radio frame, and the pattern from phase 0 up to the phase for 36864 radio frames is repeated.]~~.
<Editor's note: .there is a choice between the ETSI and ARIB inputs. ETSI supports 10 ms period but ARIB supports 2^9 superframes period. Ad Hoc 10 recommends using 10ms.>
Let $n_{40} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i*:th symbol of the sequence x_n and *y*, respectively
The *m*-sequences x_n and *y* are constructed as:
Initial conditions:
$x_n(0)=n_0$, $x_n(1)= n_1$, … $=x_n(39)= n_{39}$, $x_n(40)= n_{40}$
$y(0)=y(1)= \ldots =y(39)= y(40)=1$
Recursive definition of subsequent symbols:
$x_n(i+41) =x_n(i+3) + x_n(i)$ *modulo 2, i=0,…,* 2^{41}*-43,*
$y(i+41) = y(i+20)+y(i)$ *modulo 2, i=0,…,* 2^{41}*-43.*
The definition of the *n*:th scrambling code word for the in phase and quadrature components follows as (the left most index correspond to the chip scrambled first in each radio frame):
$c_{1,long,n} = < x_n(0)+y(0), x_n(1)+y(1), \ldots, x_n([36864x]N-1)+y([36864x]N-1) >$,
$c_{2,long,n} = < x_n(M)+y(M), x_n(M+1)+y(M+1), \ldots, x_n(M+[36864x]N-1) + y(M+[36864x]N-1) >$,
again all sums being modulo 2 additions. (Both N and M are defined in Table 2.)
<Editor's note: Choice has to be made between 10 ms period or 2^9 super frame period to decide inclusion of [ ].>
The antipodal version of $c_{scr1,n}$, the sequence $c'_{scr1,n}$, is defined as, for *i=0,…,* 2^{41}*-43,* if $c_{scr1,n}(i)=0$ then $c'_{scr1,n}(i)=1$, and if $c_{scr1,n}(i)=1$ then $c'_{scr1,n}(i)=-1$.

~~The code generator must be able to generate the sequence shifted arbitrarily from the initial state.~~
<Editor's note: It has been suggested in Ad Hoc 10 to remove this restriction.>
~~*<Editor's note: it has been noted as desirable to be able to lift this restriction.>*~~

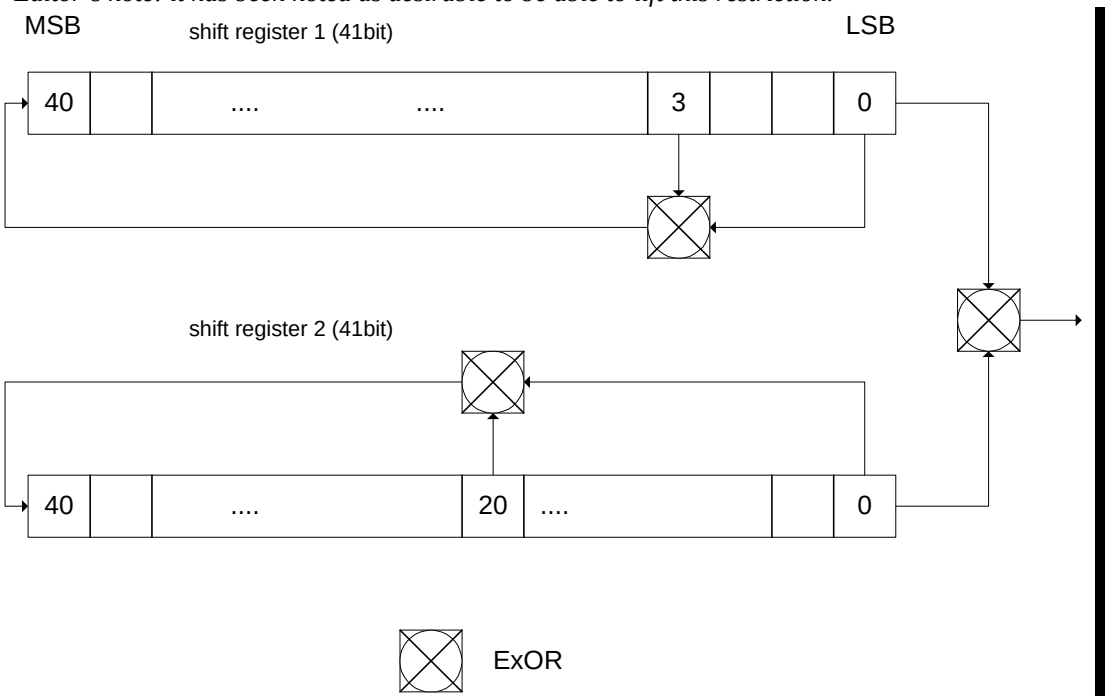


Figure 5. Configuration of uplink scrambling code generator

~~<Editor's note: the above figure is a reconstruction of the ARIB figure due to Word problems.>~~

Chip rate (Mcps)	Period N (chips)	I/Q Offset M (chips)	Range of phase (chip)	
			Primary scrambling code phase (c_1)	Secondary scrambling code phase (c_2')
[1.024	10240	896	0 – [36864x]N-1	M –[36864x]N+(M-1)
4.096	40960	3584		
[8.192	81920	7168		
[16.384	163840	14336		

Table 2. Correspondence between chip rate and uplink scrambling code phase range

<Editor's note: 1.024Mcps is not defined in ETSI document. The chip rate of 8.192and 16.384Mcps are ffs at ETSI.>

7.3.2.2 Short scrambling code

<Editor's note: Ad Hoc 10 has recommended that Short scrambling is not included in Release 99. If this is ratified the short scrambling section will be removed to a suitable place FFS.>

The short scrambling codes are formed as described in Section 6.3.2.1, where, c_1 and c_2 are two different codes from the [extended Very Large Kasami set of length 256] [periodic extended S(2) code family].

<Editor's note:.there is a choice between the ETSI and ARIB inputs. The ETSI scheme has no detailed description for VL Kasami. The following is the ARIB S(2) text.>

[

The reverse link short codes $S_v(n)$, n=0,1,…255, of length 256 chips are obtained by one chip periodic extension of S(2) sequences of length 255. It means that the first chip ($S_v(0)$) and the last chip ($S_v(255)$) of any reverse short scrambling code are the same.

The quaternary S(2) sequence $z_v(n)$, $0 \le v \le 16{,}777{,}216$, of length 255 is obtained by modulo 4 addition of three sequences, a quaternary sequence $a_r(n)$ and two binary sequences $b_s(n)$ and $c_t(n)$, according to the following relation:

$$z_v(n) = a_r(n) + 2 \cdot b_s(n) + 2 \cdot c_t(n) \quad (mod\ 4)\ , \quad n = 0, 1, \ldots, 254.$$

The user index v determines the indexes r, s, and t of the constituent sequences in the following way:

$$v = t \cdot 2^{16} + s \cdot 2^8 + r\ ,$$

r = 0, 1, 2, …, 255,
s = 0, 1, 2, …, 255,
t = 0, 1, 2, …, 255.

The quaternary sequence $a_r(n)$ is generated by the recursive generator G_0 defined by the polynomial $g_0(x)= x^8+x^5+3x^3+x^2+2x+1$ as

~~$a_r(n)= a_r(n-3) 3 a_r(n-5) a_r(n-6) 2 a_r(n-7) a_r(n-8) (mod 4).$~~
$a_r(n)= 3.a_r(n-3) +1.a_r(n-5) + 3.a_r(n-6) +2.a_r(n-7) + 3.a_r(n-8) \pmod 4.$
<Editor's note: this form of the equation has the same meaning as the previous form but is more consistent with the figure below.>

The binary sequence $b_s(n)$ is generated by the recursive generator G_1 defined by the polynomial $g_1(x)= x^8+x^7+x^5+x+1$ as

$b_s(n)= b_s(n-1)+ b_s(n-3)+ b_s(n-7)+ b_s(n-8) \pmod 2$.

The binary sequence $c_t(n)$ is generated by the recursive generator G_2 defined by the polynomial $g_2(x)= x^8+x^7+x^5+x^4+1$ as

$c_t(n)= c_t(n-1)+ c_t(n-3)+ c_t(n-4)+ c_t(n-8) \pmod 2$.

An implementation of the short scrambling code generator is shown in Figure 6. The initial states for the binary generators G_1 and G_2 are the two 8-bit words representing the indexes s and t in the 24-bit binary representation of the user index v, as it is shown in Figure 7.
The initial state for the quaternary generator G_0 is according to Figure 7. obtained after the transformation of 8-bit word representing the index r. This transformation is given by

$a_r(0) = 2v(0)+1 \pmod 4$, $a_r(n) = 2v(n) \pmod 4$, $n = 1,\ldots,7$.

The complex quadriphase sequence $S_v(n)$ is obtained from quaternary sequence $z_v(n)$ by the mapping function given in Table 3.
The Re{Sv(n)} and Im{Sv(n)} of the S(2) code are the pair of two binary sequences corresponding to input binary sequences c_1 and c_2 respectively described in 6.3.2.
.

zv(n)	Sv(n)
0	+1 + j1
1	-1 + j1
2	-1 - j1
3	+1 - j1

Table 3. Mapping between $S_v(n)$ and $z_v(n)$

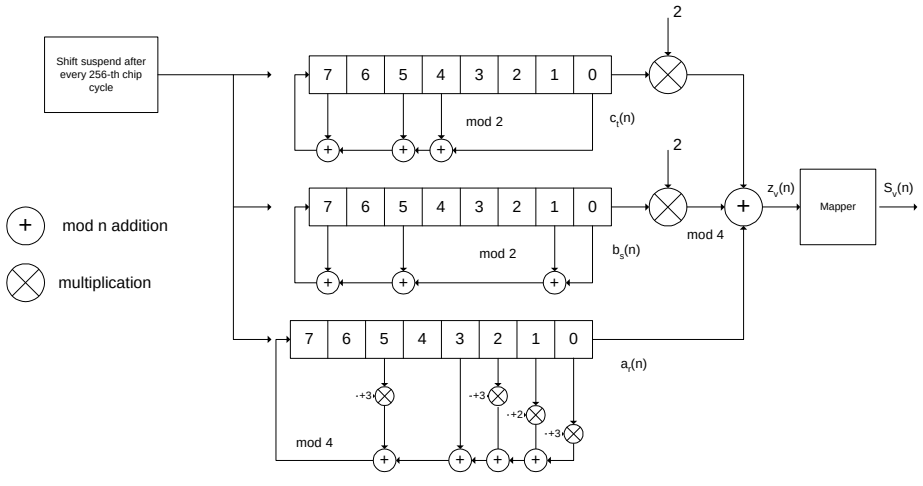


Figure 6. Uplink short scrambling code generator

<editor's note: the above figure has been re-drawn for clarity without changing the meaning.>

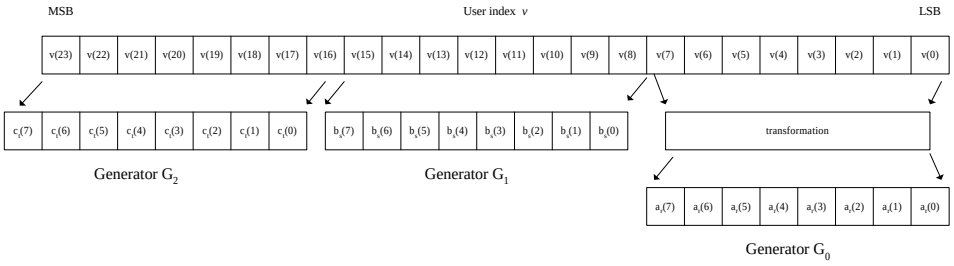


Figure 7. Uplink short scrambling code generator state initialisation

]

The uplink short scrambling code is decided by the network. The mobile station is informed about what short scrambling code to use in the downlink Access Grant message that is the base-station response to an uplink Random Access Request.

The short scrambling code may, in rare cases, be changed during a connection.

7.3.3 Random access codes

<Editor's node: ARIB terminology for PRACH is Reverse Link Common Physical Channel. ARIB has no preamble and the equivalent of the message part has "data" on the I-phase and "signature" on the Q-phase. This is similar to the ETSI message part having data part on I and control part (pilot and rate information) on Q.>

7.3.3.1 Preamble spreading code

The spreading code for the preamble part is cell specific and is broadcast by the base station. More than one preamble code can be used in a base station if the traffic load is high. The preamble codes must be code planned, since two neighbouring cells should not use the same preamble code.

The code used is a real-valued 256 chip Orthogonal Gold code. All 256 codes are used in the system.

The code sequences are constructed with the help of two binary *m*-sequences of length 255, *x*, and *y*, respectively. The *x* sequence is constructed using the polynomial $1+X^2+X^3+X^4+X^8$. The *y* sequence is constructed using the polynomial $1+X^3+X^5+X^6+X^8$.

Let $n_7 \ldots n_0$ be the binary representation of the code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen code number *n* and is denoted x_n in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y*, respectively

The *m*-sequences x_n and *y* are constructed as:

Initial conditions:

$x_n(0)=n_0$, $x_n(1)= n_1$, $\ldots = x_n(6)= n_6$, $x_n(7)= n_7$

$y(0)=y(1)= \ldots =y(6)= y(7)=1$

Recursive definition of subsequent symbols:

$x_n(i+8) =x_n(i+4) + x_n(i+3) + x_n(i+2) + x_n(i)$ *modulo 2, i=0,…,* 246,

$y(i+8) = y(i+6)+ y(i+5)+ y(i+3)+y(i)$ *modulo 2, i=0,…,* 246.

The definition of the *n*:th code word follows (the left most index correspond to the chip transmitted first in each slot):

$C_{RACH,n} = < 0, x_n(0)+y(0), x_n(1)+y(1), \ldots, x_n(254)+y(254) >$,

All sums of symbols are taken modulo 2.

The preamble spreading~~scambling~~ code is described in Figure 8.

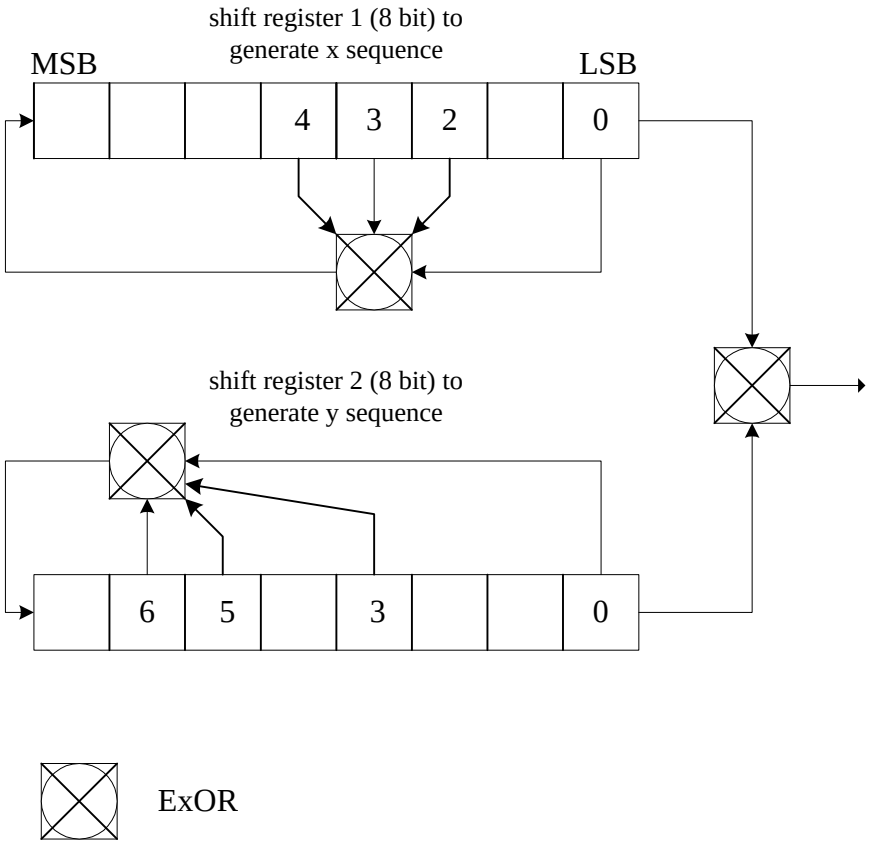


Figure 8. Preamble spreadings~~scrambling~~ code generator

Note that the code words always start with a constant ‘0’. symbol.
Before modulation and transmission these binary code words are converted to real valued sequences by the transformation ‘0’ -> ‘+1’, ‘1’ -> ‘-1’.

7.3.3.2 Preamble signature

The preamble part carries one of 16 different orthogonal complex signatures of length 16, $<P_0, P_1, ..., P_{15}>$. The signatures are based on a set of Orthogonal Gold codes of length 16 and are specified in Table 4. The base station broadcasts which signatures are allowed to be used in a cell.

	Preamble symbols															
Signature	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
1	A	A	A	-A	-A	-A	A	-A	-A	A	A	-A	A	-A	A	A
2	-A	A	-A	-A	A	A	A	-A	A	A	A	-A	-A	A	-A	A
3	A	-A	A	A	A	-A	A	A	-A	A	A	A	-A	A	-A	A
4	-A	A	-A	A	-A	-A	-A	-A	-A	A	-A	A	-A	A	A	A
5	A	-A	-A	-A	-A	A	A	-A	-A	-A	-A	A	-A	-A	-A	A
6	-A	-A	A	-A	A	-A	A	-A	A	-A	-A	A	A	A	A	A
7	-A	A	A	A	-A	-A	A	A	A	-A	-A	-A	-A	-A	-A	A
8	A	A	-A	-A	-A	-A	-A	A	A	-A	A	A	A	A	-A	A
9	A	-A	A	-A	-A	A	-A	A	A	A	-A	-A	-A	A	A	A
10	-A	A	A	-A	A	A	-A	A	-A	-A	A	A	-A	-A	A	A
11	A	A	A	A	A	A	-A	-A	A	A	-A	A	A	-A	-A	A
12	A	A	-A	A	A	A	A	A	-A	-A	-A	-A	A	A	A	A
13	A	-A	-A	A	A	-A	-A	-A	A	-A	A	-A	-A	-A	A	A
14	-A	-A	-A	A	-A	A	A	A	A	A	A	A	A	-A	A	A
15	-A	-A	-A	-A	A	-A	-A	A	-A	A	-A	-A	A	-A	-A	A
16	-A	-A	A	A	-A	A	-A	-A	-A	-A	A	-A	A	A	-A	A

Table 4. Preamble signatures. A = 1+j.

7.3.3.3 Channelization codes for the message part

The signature in the preamble specifies one of the 16 nodes in the code-tree that corresponds to channelization codes of length 16, as shown in Figure 9. The sub-tree below the specified node is used for spreading of the message part. The control (Q-branch) is spread with the channelization code of spreading factor 256 in the lowest branch of the sub-tree. The data part (I-branch) can use any of the channelization codes from spreading factor 32 to 256 in the upper-most branch of the sub-tree. However, the system may restrict the set of codes (spreading factors) actually allowed in the cell, through the use of a BCH message.

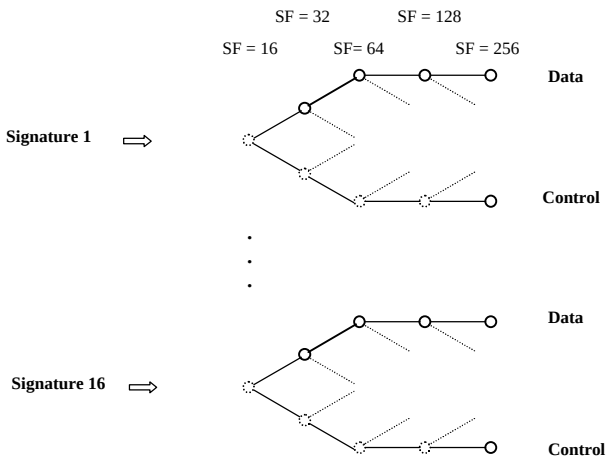


Figure 9. Channelization codes for the random access message part.

Since the control part is always spread with a known channelization code of length 256, it can be detected by the base station. The rate information field of the control part informs the base station about the spreading factor used on the data part. With knowledge of the sub-tree (obtained from the preamble signature) and the spreading factor (obtained from the rate information), the base station knows which channelization code is used for the data part.

This structure allows for simultaneous detection of multiple random access messages arriving in the same access slot, as long as different signatures are used.

7.3.3.4 Scrambling code for the message part

In addition to spreading, the message part is also subject to scrambling with a 10 ms complex code. The scrambling code is cell-specific and has a one-to-one correspondence to the spreading code used for the preamble part. Note that although the scrambling code is the same for every access slot, there is no scrambling-code collision problems between different access slots due to the 1.25 ms time shifts between the access slots.

~~*<Editor's note: the 10ms length for the scrambling code is from ETSI. The ARIB reverse link scrambling scheme has a sequence length of 36,864 frames each of 10 ms. The decision taken for uplink scrambling codes has an effect on the PRACH scrambling codes.>*~~

The scrambling codes used are from the same set of codes as is used for the other dedicated uplink channels when the long scrambling codes are used for these channels. The first 256 of the long scrambling codes are used for the random access channel. The generation of these codes is explained in Section Long scrambling code. The mapping of these codes to provide a complex scrambling code is also the same as for the other dedicated uplink channels and is described in Section **Error! Not a valid bookmark self-reference.**.

7.4 Modulation

7.4.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps. *<Editor's note: 1.024 is in ARIB but not ETSI.>*

7.4.2 Pulse shaping

The pulse-shaping filters are root-raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

7.4.3 Modulation

In the uplink, the data modulation of both DPCCH and DPDCH is BPSK. The modulated DPCCH is mapped to the Q-channel, while the first DPDCH is mapped to the I-channel. Subsequently added DPDCHs are mapped alternatively to the I or Q-channel.

8 Downlink spreading and modulation

8.1 Spreading

Figure 10 illustrates the spreading and modulation for the downlink DPCH and CCPCHs. Data modulation is QPSK where each pair of two bits are serial-to-parallel converted and mapped to the I and Q branch respectively. The I and Q branch are then spread to the chip rate with the same channelization code c_{ch} (real spreading) and subsequently scrambled by the same cell specific scrambling code C_{scramb} (complex scrambling).

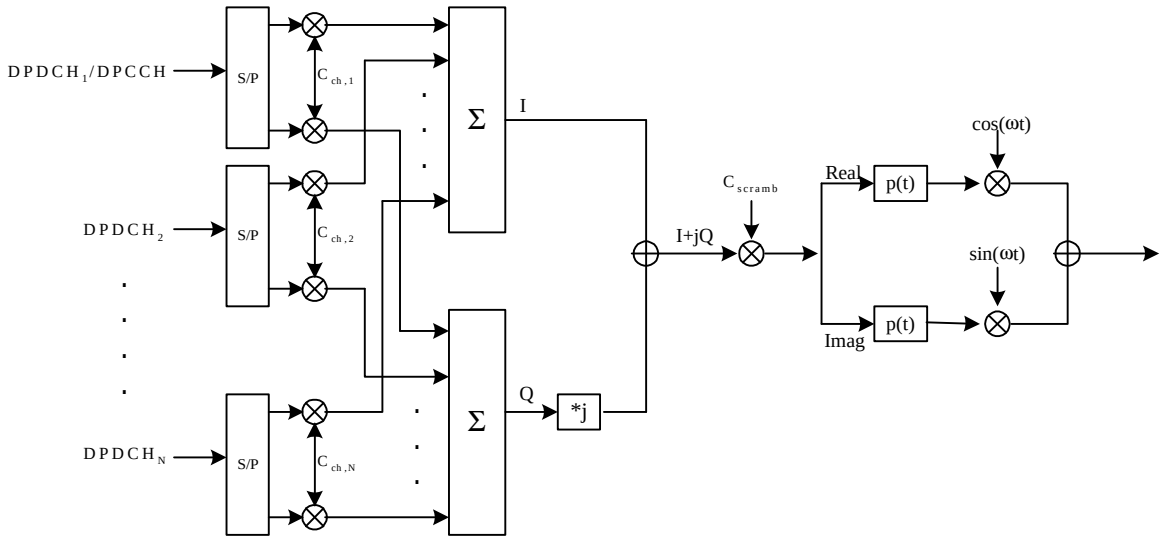


Figure 10. Spreading/modulation for downlink DPCH and CCPCHs.

The different physical channels use different channelization codes, while the scrambling code is the same for all physical channels in one cell.

The multiplexing of the [SCH]{1st and 2nd search codes} with [Primary CCPCH] {Perch channel} is illustrated in Figure 11. The figure illustrates that the [SCH]{1st and 2nd search code} are transmitted alternately. [Primary SCH and Secondary SCH] {The first search code and the second search code} are code multiplexed and transmitted simultaneously during the 1st symbol (256 chips) of each slot. In a frame. The transmission power of [SCH] {search code symbol} can be adjusted by a gain factor $G_{P\text{-}SCH}$ and $G_{S\text{-}SCH}$, respectively, independent of transmission power of P-CCPCH. The [SCH]{search code} is *non-orthogonal* to the other downlink physical channels.

[Editor's note: Take texts in [ ] if ETSI's SCH is chosen. Take texts in { } if ARIB's perch channel scheme is chosen.]

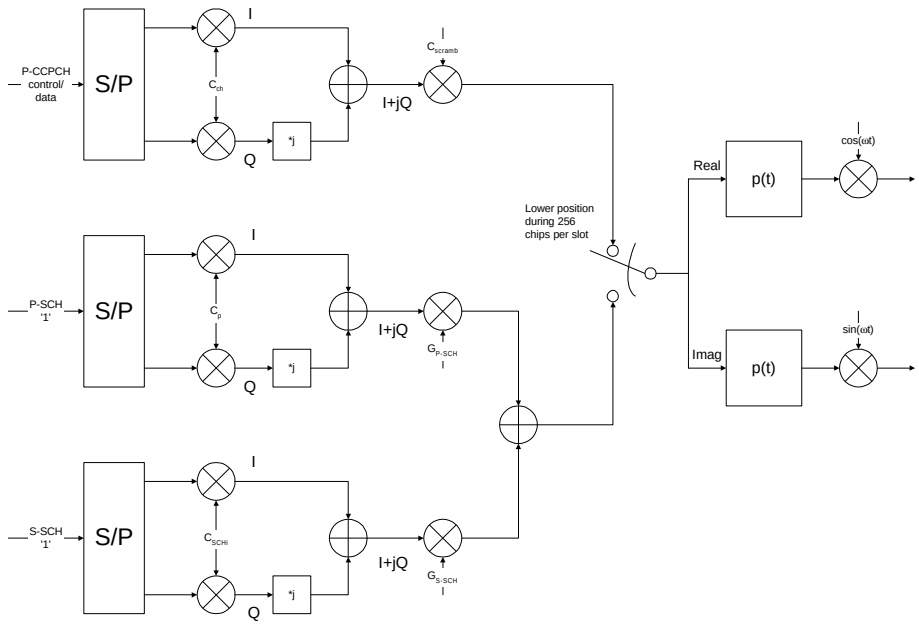


Figure 11. [Multiplexing of SCH]{Transmission of PCCPCH}

<Editor's note: the optional scale parameters reflects the suggestion from the Ad Hoc that CCPCH and SCH channels need not have the same amplitude. If this is not adopted the scale factor will be unity.>~~<Editor's note: The above two figures should be drawn in a more consistent way, maybe as one figure only. >~~

8.2 Code generation and allocation

8.2.1 Channelization codes

The channelization codes of Figure 10 are the same codes used in the uplink, namely Orthogonal Variable Spreading Factor (OVSF) codes that preserve the orthogonality between downlink channels of different rates and spreading factors. The OVSF codes are defined in Figure 3 in Section Channelization codes. The same restriction on code allocation applies as for the uplink, but for a cell and not a mobile station as in the uplink. Hence, in the downlink a code can be used in a cell if and only if no other code on the path from the specific code to the root of the tree or in the sub-tree below the specific code is used in the same cell.

The channelization code for the BCH is a predefined code which is the same for all cells within the system.

The channelization code(s) used for the [Secondary Common Control Physical Channel][Common Physical Channel] is broadcast on the BCH.

< Editor's note: There is a choice in terminology.>

The channelization codes for the downlink dedicated physical channels are decided by the network. The mobile station is informed about what downlink channelization codes to receive in the downlink Access Grant message that is the base-station response to an uplink Random Access request. The set of channelization codes may be changed during the duration of a connection, typically as a result of a change of service or an inter-cell handover. A change of downlink channelization codes is negotiated over a DCH.

8.2.2 Scrambling code

The total number of available scrambling codes is 512, divided into 32 code groups with 16 codes in each group. The grouping of the downlink codes is done in order to facilitate a fast cell search, see document S1.14. The downlink scrambling code is assigned to the cell (sector) at the initial deployment. The mobile station learns about the downlink scrambling code during the cell search process, see document S1.14.

~~*<Editor's note: this dependency on S1.14 should be verified during harmonisation>.*~~

[In order to avoid code limitation in some cases, e.g. when increasing the capacity using adaptive antennas, the possibility to associate several scrambling codes with one cell (BCH area) has been identified as one solution. The exact implementation of such a scheme is still to be determined.]

<Editor's note: Use of multiple downlink scrambling codes to aid adaptive antennas are not specified in ARIB document and treated as ffs.>

The scrambling code sequences are constructed by combining two real sequences into a complex sequence. Each of the two real sequences are constructed as the position wise modulo 2 sum of [40960 chip segments of] two binary *m*-sequences generated by means of two generator polynomials of degree 18. The resulting sequences thus constitute segments of a set of Gold sequences. The scrambling codes are repeated for every 10 ms radio frame. Let *x* and *y* be the

two sequences respectively. The *x* sequence is constructed using the primitive (over GF(2)) polynomial $1+X^7+X^{18}$. The y sequence is constructed using the polynomial $1+X^5+X^7+X^{10}+X^{18}$.
*<Editor's note: [ ] is due to the fact that only 4.096Mcps is defined in the ETSI documents. ARIB documents also defines 1.024, 8.196, and 16.384Mcps.>*Let $n_{17} \ldots n_0$ be the binary representation of the scrambling code number *n* (decimal) with n_0 being the least significant bit. The *x* sequence depends on the chosen scrambling code number *n* and is denoted x_n, in the sequel. Furthermore, let $x_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence x_n and *y*, respectively
The *m*-sequences x_n and *y* are constructed as:
Initial conditions:
$x_n(0)=n_0$, $x_n(1)= n_1$, $\ldots = x_n(16)= n_{16}$, $x_n(17)= n_{17}$
$y(0)=y(1)= \ldots =y(16)= y(17)=1$
Recursive definition of subsequent symbols:
$x_n(i+18) =x_n(i+7) + x_n(i)$ *modulo 2,* $i=0,\ldots,2^{18}-20$,
$y(i+18) = y(i+10)+y(i+7)+y(i+5)+y(i)$ *modulo 2,* $i=0,\ldots, 2^{18}-20$.
The n:th Gold code sequence z_n is then defined as
$z_n(i) = x_n(i) + y(i)$ *modulo 2,* $i=0,\ldots, 2^{18}-2$.
The antipodal version of z_n, the sequence z'_n, is defined as, for $i=0,\ldots, 2^{18}-2$, if $z_n(i)=0$ *then* $z'_n(i)=1$, *and if* $z_n(i)=1$ *then* $z'_n(i)=-1$.
Finally, the n:th complex scrambling code sequence C_{scramb} is defined as (the lowest index corresponding to the chip scrambled first in each radio frame): (see Table 5 for definition of N and M)

$C_{scramb}(i) = z'_n(i) + j\, z'_n(i+M)$, $i=0,1,\ldots,N-1$.
<Editor's note: the values 3584 and 40960 are based on an assumption of a chip rate of 4.096 Mcps.>
Note that the pattern from phase 0 up to the phase of 10 msec is repeated.
The index *n* runs from 0 to 511 giving 512 distinct 40960 chip sequences.
<Editor's note: in ARIB the generator is required to be able to realise a clock arbitrarily shifted from the initial phase. It has been suggested in Ad Hoc 10 to not include this restriction.>

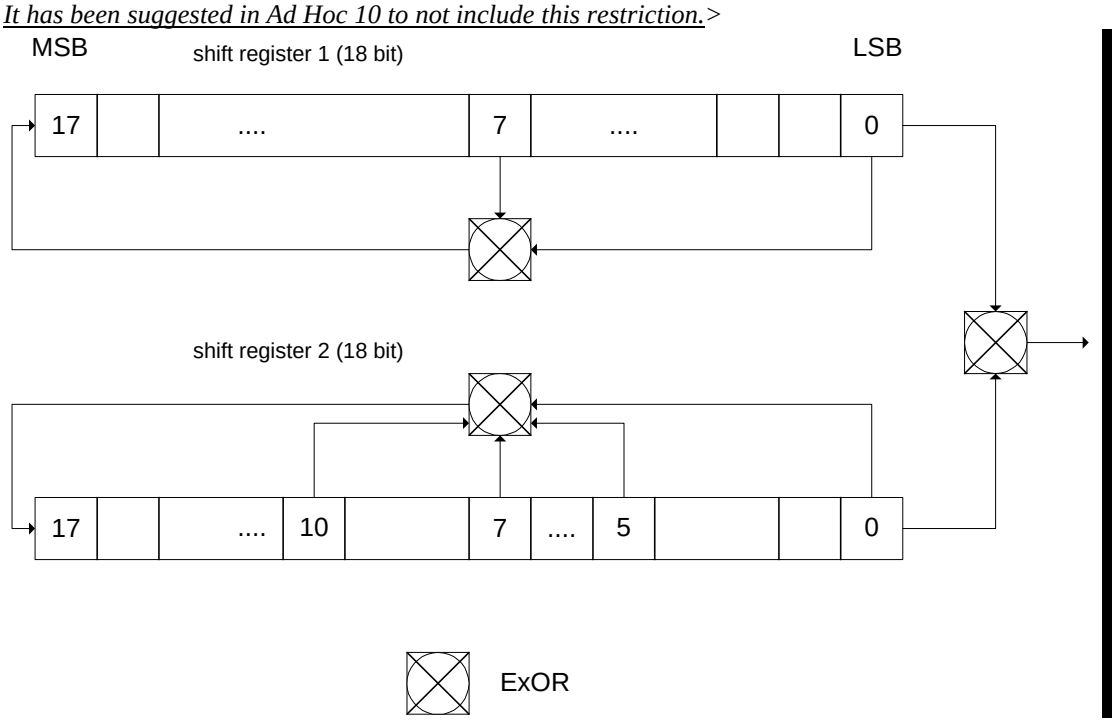


Figure 12. Configuration of downlink scrambling code generator

<Editor's note: the above figure is a reconstruction of the ARIB figure due to Word problems.>

chip rate (Mcps)	Period N	IQ Offset M	Range of phase (chip)	
			for in-phase component	for quadrature component
[1.024	[10240]	[896]	0 – N-1	M – N+M-1
4.096	40960	3584		
[8.192	[81920]	[7168]		
[16.384	[163840]	[14336]		

Table 5. Correspondence between chip rate and downlink scrambling code phase range

8.2.3 Synchronisation codes

8.2.3.1 Code Generation

The Primary and Secondary code words, C_p and $\{C_1,\ldots,C_{17}\}$ are constructed as the position wise addition modulo 2 of a Hadamard sequence and a fixed so called hierarchical sequence. The [Primary SCH] [1st search code] is furthermore chosen to have good aperiodic auto correlation properties.

< Editor's note: There is a choice on the terminology. Also, the text in the 2nd [ ] needs to be verified>

The hierarchical sequence y sequence is constructed from two constituent sequences x_1 and x_2 of length n_1 and n_2 respectively using the following formula:

$y(i) = x_2(i \bmod n_2) + x_1(i \operatorname{div} n_2)$ *modulo 2*, $i = 0 \ldots (n_1 * n_2) - 1$

The constituent sequences x_1 and x_2 are chosen to be identical and to be the following length 16 (i.e. $n_1 = n_2 = 16$) sequence:

$x_1 = x_2 = < 0, 0, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0 >$

The Hadamard sequences are obtained as the rows in a matrix H_8 constructed recursively by:

$$H_0 = (0)$$

$$H_k = \begin{pmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & \overline{H_{k-1}} \end{pmatrix}, \quad k \geq 1$$

The rows are numbered from the top starting with row *0* (the all zeros sequence).

The Hadamard sequence *h* depends on the chosen code number *n* and is denoted h_n in the sequel.

[This code number is chosen from every 8th low of the matrix H_8. Therefore, there are 32 possible code numbers out of which 17 are used.]

<Editor's note: Only ARIB input specifies this code group out of which 17 codes are chosen but it has to do with fast Hadamard transformation>

Furthermore, let $h_n(i)$ and $y(i)$ denote the *i:*th symbol of the sequence h_n and *y*, respectively.

$h(i)$ is identical to $C_{2^8,i'}$ where i' is the bit-reversed number of the 8-bit binary representation of i.

The definition of the *n*:th [SCH][search] code word follows (the left most index correspond to the chip transmitted first in each slot):

$C_{SCH,n} = < h_n(0)+y(0), h_n(1)+y(1), h_n(2)+y(2), \ldots, h_n(255)+y(255) >$,

All sums of symbols are taken modulo 2.

Before modulation and transmission these binary code words are converted to real valued sequences by the transformation '0' -> '+1', '1' -> '-1'.

The [Primary SCH][1st] search] and [Secondary SCH][2nd search] code words are defined in terms of $C_{SCH,n}$ and the definition of C_p and $\{C_1,\ldots,C_{17}\}$ now follows as:

$C_p = C_{SCH,\,0}$

and

$C_i = C_{SCH,\,i}$, i=1,…,17

The definitions of C_p and $\{C_1,\ldots,C_{17}\}$ are such that a 32 point fast Hadamard transform can be utilised for detection.

<Editor's note: choice has to be made between for example primary SCH code and 1st search code.>

8.2.3.2 Code Allocation

<Editor's note: Descriptions in ETSI's XX.03 5.3.2.3 and in ARIB's 3.2.4.2.3.1.3.1. can be found here.>

<Editor's note: Following text is copied in large part out of 5.3.2.3 of S1.11 with insertion of text taken out of ARIB document where appropriate as the code allocation of Primary and Secondary SCH channels (or 1st and 2nd search code according to ARIB terminology) is more suitably found in this section. (See editorial comment in 5.3.2.3 of S1.11>

The 32 sequences are constructed such that their cyclic-shifts are unique, i.e., a non-zero cyclic shift less than 16 of any of the 32 sequences is not equivalent to some cyclic shift of any other of the 32 sequences. Also, a non-zero cyclic shift less than 16 of any of the sequences is not equivalent to itself with any other cyclic shift less than 16. This property is

used to uniquely determine both the long code group and the frame timing in the second step of acquisition (see S1.14 Section 4.1). The following sequences are used to encode the 32 different code groups each containing 16 scrambling codes (note that c_i indicates the i'th Secondary Short code [2nd search code] of the 17 codes). Note that a Secondary Short code [2nd search code] can be different from one time slot to another and that the sequence pattern can be different from one cell to another, depending on Scrambling Code Group of Scrambling Code the cell uses~~. [In this table, the last pattern (33rd : Sync BTS) is assigned to the synchronous base station operations, in which only one Scrambling Code is used.]~~

~~*<Editor's note: the 33rd sequence in the table below is used for detecting synchronous CDMA systems such as cdma2000. As such, the 33rd sequence is only found in ARIB document.>*~~

Scrambling Code Groups	Slot Number															
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16
Group1	C_1	C_1	C_2	C_{11}	C_6	C_3	C_{15}	C_7	C_8	C_8	C_7	C_{15}	C_3	C_6	C_{11}	C_2
Group2	C_1	C_2	C_9	C_3	C_{10}	C_{11}	C_{13}	C_{13}	C_{11}	C_{10}	C_3	C_9	C_2	C_1	C_{16}	C_{16}
Group 3	C_1	C_3	C_{16}	C_{12}	C_{14}	C_2	C_{11}	C_2	C_{14}	C_{12}	C_{16}	C_3	C_1	C_{13}	C_4	C_{13}
Group 4	C_1	C_4	C_6	C_4	C_1	C_{10}	C_9	C_8	C_{17}	C_{14}	C_{12}	C_{14}	C_{17}	C_8	C_9	C_{10}
Group 5	C_1	C_5	C_{13}	C_{13}	C_5	C_1	C_7	C_{14}	C_3	C_{16}	C_8	C_8	C_{16}	C_3	C_{14}	C_7
Group 6	C_1	C_6	C_3	C_5	C_9	C_9	C_5	C_3	C_6	C_1	C_4	C_2	C_{15}	C_{15}	C_2	C_4
Group 7	C_1	C_7	C_{10}	C_{14}	C_{13}	C_{17}	C_3	C_9	C_9	C_3	C_{17}	C_{13}	C_{14}	C_{10}	C_7	C_1
Group 8	C_1	C_8	C_{17}	C_6	C_{17}	C_8	C_1	C_{15}	C_{12}	C_5	C_1	C_7	C_{13}	C_5	C_{12}	C_{15}
Group 9	C_1	C_9	C_7	C_{15}	C_4	C_{16}	C_{16}	C_4	C_{15}	C_7	C_9	C_1	C_{12}	C_{17}	C_{17}	C_{12}
Group 10	C_1	C_{10}	C_{14}	C_7	C_8	C_7	C_{14}	C_{10}	C_1	C_9	C_5	C_{12}	C_{11}	C_{12}	C_5	C_9
Group 11	C_1	C_{11}	C_4	C_{16}	C_{12}	C_{15}	C_{12}	C_{16}	C_4	C_{11}	C_1	C_6	C_{10}	C_7	C_{10}	C_6
Group 12	C_1	C_{12}	C_{11}	C_8	C_{16}	C_6	C_{10}	C_5	C_7	C_{13}	C_{14}	C_{17}	C_9	C_2	C_{15}	C_3
Group 13	C_1	C_{13}	C_1	C_{17}	C_3	C_{14}	C_8	C_{11}	C_{10}	C_{15}	C_{10}	C_{11}	C_8	C_{14}	C_3	C_{17}
Group 14	C_1	C_{14}	C_8	C_9	C_7	C_5	C_6	C_{17}	C_{13}	C_{17}	C_6	C_5	C_7	C_9	C_8	C_{14}
Group 15	C_1	C_{15}	C_{15}	C_1	C_{11}	C_{13}	C_4	C_6	C_{16}	C_2	C_2	C_{16}	C_6	C_4	C_{13}	C_{11}
Group 16	C_1	C_{16}	C_5	C_{10}	C_{15}	C_4	C_2	C_{12}	C_2	C_4	C_{15}	C_{10}	C_5	C_{16}	C_1	C_8
Group 17	C_1	C_{17}	C_{12}	C_2	C_2	C_{12}	C_{17}	C_1	C_5	C_6	C_{11}	C_4	C_4	C_{11}	C_6	C_5
Group 18	C_2	C_8	C_{11}	C_{15}	C_{14}	C_1	C_4	C_{10}	C_{10}	C_4	C_1	C_{14}	C_{15}	C_{11}	C_8	C_2
Group 19	C_2	C_9	C_1	C_7	C_1	C_9	C_2	C_{16}	C_{13}	C_6	C_{14}	C8	C_{14}	C_6	C_{13}	C_{16}
Group 20	C_2	C_{10}	C_8	C_{16}	C_5	C_{17}	C_{17}	C_5	C_{16}	C_8	C_{10}	C_2	C_{13}	C_1	C_1	C_{13}
Group 21	C_2	C_{11}	C_{15}	C_8	C_9	C_8	C_{15}	C_{11}	C_2	C_{10}	C_6	C_{13}	C_{12}	C_{13}	C_6	C_{10}
Group 22	C_2	C_{12}	C_5	C_{17}	C_{13}	C_{16}	C_{13}	C_{17}	C_5	C_{12}	C_2	C_7	C_{11}	C_8	C_{11}	C_7
Group 23	C_2	C_{13}	C_{12}	C_9	C_{17}	C_7	C_{11}	C_6	C_8	C_{14}	C_{15}	C_1	C_{10}	C_3	C_{16}	C_4
Group 24	C_2	C_{14}	C_2	C_1	C_4	C_{15}	C_9	C_{12}	C_{11}	C_{16}	C_{11}	C_{12}	C_9	C_{15}	C_4	C_1
Group 25	C_2	C_{15}	C_9	C_{10}	C_8	C_6	C_7	C_1	C_{14}	C_1	C_7	C_6	C_8	C_{10}	C_9	C_{15}
Group 26	C_2	C_{16}	C_{16}	C_2	C_{12}	C_{14}	C_5	C_7	C_{17}	C_3	C_3	C_{17}	C_7	C_5	C_{14}	C_{12}
Group 27	C_2	C_{17}	C_6	C_{11}	C_{16}	C_5	C_3	C_{13}	C_3	C_5	C_{16}	C_{11}	C_6	C_{17}	C_2	C_9
Group 28	C_2	C_1	C_{13}	C_3	C_3	C_{13}	C_1	C_2	C_6	C_7	C_{12}	C_5	C_5	C_{12}	C_7	C_6
Group 29	C_2	C_2	C_3	C_{12}	C_7	C_4	C_{16}	C_8	C_9	C_9	C8	C_{16}	C_4	C_7	C_{12}	C_3
Group 30	C_2	C_3	C_{10}	C_4	C_{11}	C_{12}	C_{14}	C_{14}	C_{12}	C_{11}	C_4	C_{10}	C_3	C_2	C_{17}	C_{17}
Group 31	C_2	C_4	C_{17}	C_{13}	C_{15}	C_3	C_{12}	C_3	C_{15}	C_{13}	C_{17}	C_4	C_2	C_{14}	C_5	C_{14}
Group 32	C_2	C_5	C_7	C_5	C_2	C_{11}	C_{10}	C_9	C_1	C_{15}	C_{13}	C_{15}	C_1	C_9	C_{10}	C_{11}
[SyncBTS]	C_2	C_6	C_{14}	C_{14}	C_6	C_2	C_8	C_{15}	C_4	C_{17}	C_9	C_9	C_{17}	C_4	C_{15}	C_8
~~[SyncBTS]~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~	~~C_1~~

Table 9 Spreading Code allocation for Secondary SCH [Second Search] Code
<Editor's note: The ARIB document assigns the sequence number to each time slot but the ETSI document does not.>

8.3 Modulation

8.3.1 Modulating chip rate

The modulating chip rate is 4.096 Mcps. This basic chip rate can be extended to [1.024,]8.192 or 16.384 Mcps.
<Editor's note: 1.024 is in ARIB but not ETSI.>

8.3.2 Pulse shaping

The pulse-shaping filters are root raised cosine (RRC) with roll-off α=0.22 in the frequency domain.

8.3.3 Modulation

QPSK modulation is used.

9 History

Document history		
draft	1999-02-12	New document merged from ETSI XX.05 and ARIB 3.2.4 sources.
0.0.1	1999-02-12	Corrected typo in table2.
0.0.2	1999-02-16	Added sec. SCH code table, option for HPSK on S(2) codes, scale on SCH.
0.0.3	1999-02-18	Reflected decision made on SCH multiplexing (see document titled 'Report from Ad Hoc #2 SCH multiplexing'.) and additional description on the use of S(2) for uplink short scrambling code.
0.1.0	1999-02-28	Raised to 0.1.0 after TSG RAN WG1#2 meeting (Yokohama).
1.0.0	1999-03-12	Raised to 1.0.0 when presented to TSG RAN.
1.0.1	1999-03-17	Raised to 1.0.1 incorporated Ad Hoc changes and errata from e-mail.

Editor for S1.13, spreading and modulation description, is:

Peter Chambers
Siemens Roke Manor Research
Email: **Error! Bookmark not defined.**

This document is written in Microsoft Word 97.